A METHOD FOR MEASURING PROGRAMMATIC DEPENDENCY AND INTERDEPENDENCY BETWEEN DOD ACQUISITION PROGRAMS

THESIS

Matthew B Christensen, Captain, USAF

AFIT/GSE/ENV/11-D02DL

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GSE/ENV/11- D02DL

A METHOD FOR MEASURING PROGRAMMATIC DEPENDENCY AND INTERDEPENDENCY BETWEEN DOD ACQUISITION PROGRAMS

THESIS

Presented to the Faculty

Department of Systems and Engineering Management

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Systems Engineering

Matthew B Christensen, BS

Captain, USAF

December 2011

AFIT/GSE/ENV/11- D02DL

A METHOD FOR MEASURING PROGRAMMATIC DEPENDENCY AND INTERDEPENDENCY BETWEEN DOD ACQUISITION PROGRAMS

Matthew B Christensen, BS

Captain, USAF

Approved:

_____	_____
J. Robert Wirthlin, PhD., Lt Col, USAF (Chairman)	Date
_____	_____
John M. Colombi, PhD. (Member)	Date
_____	_____
Alfred E. Thal, Jr, PhD. (Member)	Date

Abstract

There is currently no accepted method for quantitative measurement of acquisition program interdependency. While some methods exist for measuring dependency or interdependency at the component or system level, these methods do not translate well to program interdependency measurement. The objective of this thesis is to provide a model for measuring acquisition program interdependency accurately and quantitatively.

The model presented in this thesis uses four Interdependency Factors to identify dependency relationships between programs. Specific Interdependency Levels are then used to measure the strengths of those dependencies. The model also accounts for measurement of dependencies upon programs that are not directly connected, i.e., programs that have a degree of separation from another program, and measurement of program criticality, or the extent to which a program is depended-upon.

In this thesis, the measurement model is applied to an example program to measure program dependency characteristics. The results demonstrate that the model can be effectively used to identify and measure program dependencies. The model gives the program manager a quantitative tool to determine how much a program depends upon other programs and the potential impacts of those dependencies. With this information, program managers can better protect their programs from vulnerabilities associated with interdependency effects from other programs.

Acknowledgments

I would like to thank my father for his continued guidance and help throughout this effort. Also, Lt Col Robb Wirthlin for his help and patience as we've worked through this; Mr. Steven J Buckley, for his unyielding tenacity in keeping things moving; and my family, for their support and love.

Matthew B. Christensen

Table of Contents

Page

List of Figures

Page

List of Tables

Page

List of Equations

Page

A METHOD FOR MEASURING PROGRAMMATIC DEPENDENCY AND INTERDEPENDENCY BETWEEN DOD ACQUISITION PROGRAMS

I. Introduction

Background

The increasing complexity and focus on joint capability of modern acquisition programs has led to a corresponding increase in the dependencies and interdependencies of acquisition programs (Brown, Flowe, & Hamel, 2007). In the current acquisition environment, a program may have major impacts on other programs based on these dependencies. At this time, there is no accepted way of quantitatively measuring these program dependencies and interdependencies, and their potential impacts on other programs.

A program may depend on other programs to ensure funding, develop technology, provide support and resources, or even justify the continued existence of the program. In these cases, the program is *dependent* upon the other program(s). In cases where two or more programs both depend on each other for their continued functioning, the programs are *interdependent.*

The objective of this thesis is to propose a method and model for measuring dependency and interdependency characteristics between acquisition programs quantitatively. Most existing methods and models for dependency and interdependency measurement have been created for application at the system or component level rather than the program level. Most existing methods also depend on expert opinion or

subjective means to measure the strengths of dependencies and interdependencies. This thesis presents a model to objectively measure program dependency and interdependency characteristics in a quantitative way.

Problem Statement

How can acquisition program dependency and interdependency characteristics be measured quantitatively in an accurate and meaningful way?

Definition of Dependency

Various definitions of dependency exist and they differ with the type of dependency. The *Merriam-Webster Dictionary* defines dependency as "the quality or state of being dependent; *especially* **:** the quality or state of being influenced or determined by or subject to another." We will adapt this definition to define program dependency as "the quality or state of a program in being influenced or determined by or subject to another program." That is, if a program is dependent, then it may be affected by another program.

We may think of dependency as a one-way characteristic. It is not a mutual quality of two entities, but rather a quality of the single dependent entity. Essentially, A depends on B, but B does not necessarily depend on A.

An example of a system-level dependency is found with the Joint Direct Attack Munition (JDAM) weapons system and the Global Positioning System (GPS) satellite constellation. JDAM is a guided, air-delivered weapon that uses GPS coordinates to precisely strike a target. GPS provides timing and location data to a variety of devices

with an appropriate receiver, including aircraft, ships, ground vehicles, handheld devices and the JDAM system. The JDAM requires data from the GPS constellation in order to successfully and precisely engage a target. Therefore, the JDAM system is dependent upon GPS. The GPS system does not, however, require JDAM in order to operate. It will continue to function whether or not a JDAM is present. Program dependencies may manifest if access to program funding, resources, or support, or program requirements are, or can be, influenced by another program.

Definition of Interdependency

The *Oxford English Dictionary* defines Interdependency and Interdependence equivalently as "The fact or condition of depending each upon the other; mutual dependence."

Interdependency is a two-way, or mutual, quality of two entities. Both entities require the other in order to function. An example at the component level is found in the components in a computer system. The hard disk, memory models, motherboard, input and output devices, and processor all require each other in order to function. None of them can function in their roles without all of the other parts functioning. The human body is another good example of a system of interdependent systems. For example, the nervous and circulatory systems require each other in order to carry out their functions. Neither can work without the other. Acquisition programs may be interdependent if they depend upon each other for resources, data, funding, requirements, etc.

Research Focus

Research will be focused primarily on Department of Defense (DoD) acquisition programs. However, the models and concepts applied and developed in this thesis should be applicable to other product development programs. The model will be used to determine the extent to which a given acquisition program depends upon other concurrent acquisition programs. It may also be used to determine the extent to which other programs depend upon the given program. The model provides a snapshot-in-time measurement of program dependencies and interdependencies, measuring the number of program dependencies and the strengths of those dependencies. The scope of the measurement may be constrained to the measurement of dependency and interdependency links between as few as two specific programs or may be expanded to include measurement between hundreds of programs. The scope may be determined and adapted by the Program Manager

Investigative Questions

The main investigative questions are given below:

1. How can program dependency and interdependency between two or more programs be measured quantitatively?
2. How can the overall dependency and interdependency characteristics for a specific program be measured quantitatively?
3. How can program vulnerabilities that are the result of program dependencies and/or interdependencies be evaluated?

4. How can we evaluate the interdependency effects of programs not directly connected to the program being measured?
5. How can program criticality that is the result of dependency and/or interdependencies by other programs be evaluated?

Methodology

The research will begin by defining the characteristics of program dependency and interdependency. The research will then propose a model to evaluate program interdependency quantitatively. The model will build upon previously developed maturity measurement models to evaluate levels of program dependency and interdependency. An exploratory case study is then presented. This study will apply the model to an acquisition program example and evaluate the appropriateness of the results.

Assumptions/Limitations

There are some aspects of system interdependency measurement which are inescapably subjective. However, the model provides guidelines and levels to limit the effects of subjective measurement. This limitation does come at a cost of precision but yields a more consistent and objective measurement.

The scope of this research will be confined to DoD acquisition programs. While the principles and models developed in this thesis may be expected to be applicable to other programs outside of the DoD, demonstration of a non-DoD application is outside the scope of this work.

The dependency and interdependency measurements from the model for a given program are valid only for the time at which the measurement is taken. As programs mature and new programs develop, dependency and interdependency relationships will continue to change. Because of these changes, the quantitative dependency and interdependency measurements for a given program represent a snapshot-in-time data point rather than a permanent program characteristic. However, the model may be assumed to be a valid tool for use at any given point in the life of a program. Therefore, an accurate measurement of program interdependency may be obtained at any time as long as the measurement is taken at that time. While future program dependency and interdependency characteristics may be estimated by using projected future program states for the measurement model, demonstration and validation of this application of the model is beyond the scope of this work.

Implications

This research gives program managers, decision-makers, and planners a quantitative tool to evaluate the criticality of a given program by determining the number of other programs dependent upon it and the strength of those dependencies. Resources can be prioritized accordingly and highly critical programs protected.

Decision-makers will also be able to determine the vulnerability of a given program to disruption as a result of the disruption of other programs. The number and strengths of dependencies for a program can be measured to determine how much the program depends on other programs. Contingency plans and concepts can then be created accordingly to minimize potential program impacts should another program or

programs be adversely affected. This ability will help to ensure continued program capacity in maintaining cost, schedule, and performance.

Preview

The literature review contains a summary of current interdependency research. The review also addresses maturity models which have been used for a variety of program and systems measurements, including interoperability and capability.

The Methodology chapter introduces and explains the program interdependency measurement model. It introduces the types of interdependencies as well as levels of program interdependency strength. The method for application of the model to an acquisition program is also presented.

The Analysis and Results chapter applies the model to an example case and demonstrates the use of the model in measuring program dependency and interdependency. The model is also used to determine the criticality of a program by measuring how strongly it is depended upon by other programs.

The Conclusions chapter summarizes the work and discusses how the model can be used in the field of DoD acquisitions. The significance and potential benefits of program interdependency measurement are highlighted. Suggestions are also given for several areas for further study.

II. Literature Review

Chapter Overview

This chapter provides an overview of the maturity model as a program measurement tool. It also reviews existing research on program interdependency, highlighting the importance of interdependency and the state of interdependency measurement.

Maturity models

A Maturity Model is, in general, a measure of the capability of an organization or entity. Maturity models usually specify several different levels of maturity starting with zero or one, immature, and moving to four or five, mature. The levels of maturity determine how adept the organization is in performing a certain task.

In 1987, the Capability Maturity Model (CMM) was developed by the Carnegie Melon Software Engineering Institute (CMU-SEI) under the sponsorship of the USAF. The CMM was useful because it provided a quasi-quantitative method to answer the question: "How good is an organization at completing a certain task?" For the initial work, the task was software development. Because this question is hard to answer in a quantitative manner, the Carnegie Melon researchers proposed levels of capability to be used in the model. The capability of the company would be evaluated against the levels and scored accordingly. The levels of the CMM ranged from 1 - Initial to 5 - Optimizing. A more capable and experienced company would be classified at a higher level and receive a higher maturity score or rating. Maturity model methods such as the CMM are

extremely useful because they transform a qualitative characteristic into a quantitative measurement. This can allow cleaner comparisons between groups or entities because it can summarize the capabilities of an organization with a single numerical value.

The LISI Model

In 1997, the MITRE Corporation developed a maturity model to measure the interoperability of information systems. This model, the Levels of Information Systems Interoperability (LISI), provided a way to determine how well an information system can interoperate with other information systems. The model is highly conceptual in nature and measures basic information system characteristics. The LISI model evaluates an information system based on four system attributes: Procedures, Applications, Infrastructure, and Data, or PAID (DoD 1998). A diagram of the LISI model is presented in Figure 1.

The LISI model was the first maturity model widely adopted for use in interoperability measurement (Ford, Colombi, & Jacques, 2009). It has since been adapted and modified many times to create other measurement models, including models for organizational interoperability in the Organizational Interoperability Maturity (OIM) model (Clark & Jones, 1999), and for the interoperability between acquisition program metrics (Shibata, 2010). LISI and its derivatives are illustrations of maturity models that have been adapted to measure characteristics other than capability maturity.

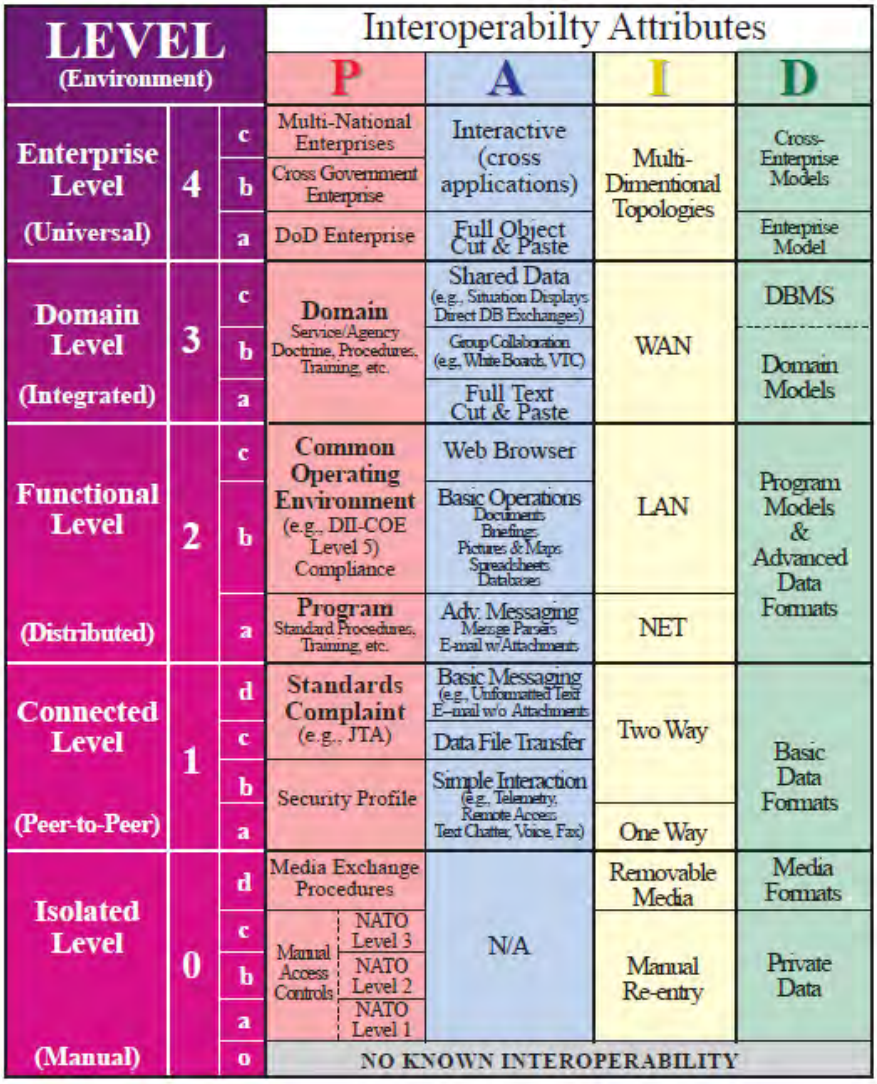


Figure 1. The LISI interoperability measurement model (DoD 1998)

Interdependency, as defined in Chapter I, is a programmatic or systematic quality. Unfortunately, qualitative statements such as –The program is very interdependent with other programs" and –Program A is not very dependent on Program B" only give us a general idea of how strong the program dependency relationships are. Qualitative assessments can be harder to use effectively than quantitative measurements. Because it

allows for the application of a quantitative measurement to a program quality, the maturity model is well-suited for measuring program interdependency. Within the context of a maturity model, the phrase "Program A has a Level 3 dependency on Program B" can tell us quite a bit about the programmatic relationship. Assuming the maturity model levels are understood by the interested parties, a Level 3 quantitative dependency measurement can carry with it a great deal of qualitative information regarding how much Program A depends upon Program B.

Summary of Current Program Interdependency Research

The field of program interdependency research is a relatively new area of study. Only recently has this field started to attract the attention of program and acquisition authorities and researchers. While various methods of interdependency measurement exist for other applications, there are few methods for program interdependency measurement. Models for measuring the interdependencies in Critical Infrastructure (CI) systems, computer networks, information systems, biological systems, and even sub-atomic particles have all been proposed or published. Unfortunately, these models do not translate well into program interdependency measurement. Yet in the increasingly complex field of DoD acquisitions, measurement of program interdependencies will continue to increase in importance (Brown, Flowe, & Hamel, 2007). The remainder of this chapter summarizes existing research and methods in program dependency and interdependency measurement.

Relevant Research:

Current program interdependency research has focused largely on how the phenomenon of program interdependency can affect other program characteristics such as risk levels and resource demands. Most of the research does not specifically focus on interdependency measurement, but usually does include quantitative representations or measurements of program interdependency.

Mane and DeLaurentis introduced the Exploratory Computational Model (later the Computational Exploratory Model or CEM) as a tool to help program managers in the procurement of systems (2009). It is used to help assess the impacts of interdependencies on program development, especially system-of-systems development. This model uses a program dependency strength measurement to help determine program risks due to interdependencies. The dependency strength measurement is given as $S(i,j)$, and is defined as "the conditional probability (uniform random probability) that system *i* has a disruption, given that system *j* (on which system *i* depends) has a disruption." The CEM has been used to analyze the expected delays to interdependent programs based on adverse interdependent effects (2010), the correlation between interdependencies and program development time and risk (DeLaurentis & Sauser, April 13, 2010), and trades between system-of-systems capability and development risk (2011). At this time however, the dependency parameter value is subjectively determined through subject expert assessment rather than through objective measurement with a measurement model or measurement equation. This limitation means that different experts may arrive at different conclusions regarding the interdependency characteristics of a given program.

Brown and Flowe et al, (2010) introduced the "Effective Nodes" parameter, N_e, as part of their research on program interdependency as a predictor for program resource demand. This parameter is a total measure of program interdependency based on links to other programs. It is determined by both the number of dependency links to other programs and the send-receive characteristics of those links. Links are categorized as send only, which we would define as dependent, receive only or depended-upon, and send-receive or interdependent. In order to evaluate the program dependency, the programs are decomposed into program elements or "nodes." Dependencies between the nodes of different programs are identified and tallied for each send-receive category. N_e can then be calculated using the following equation:

$$N_e = (N_{s/r} + 0.5N_s + 0.29N_r)\left(\frac{L_t/N_t}{1.02}\right)^{1.22}$$

Equation 1. Formulation for N_e (Flowe, et al., 2010)

In this equation, N_e is the equivalent nodes value, N_s is the number of send-only nodes for the program, N_r is the number of receive-only nodes for the program, $N_{s/r}$ is the number of send-receive nodes for the program, L_t is the total number of links for the program, and N_t is the total number of nodes for the program. The equation is based on empirical program data for Major Defense Acquisition Programs (MDAPs) from Fiscal Year 2008.

The N_e factor works extremely well in characterizing overall program interdependency as it relates to program resource demand. However, this factor does not

measure the strengths of interdependencies or different types of interdependencies outside of the send-receive characteristic. All instances of a given type of program link – send, receive, or send-receive – are assumed to be the same strength. There is no further measurement of any individual program interdependency. Because of this, it is very difficult to determine the relative importance of different programs to the program of interest. This knowledge is essential to the program manager who wishes to evaluate and protect against program vulnerabilities related to interdependency or who wishes to assess which other programs have the most potential influence over the program. In order to specifically evaluate individual interdependencies between programs, an expanded method is needed.

Finally, Asikoglu and Simpson (2010) have also done significant research in the area of product component dependency measurement. Their method assigns Module Complexity Scores (MCS) to different components based on interface types, and then uses an electrical circuit analogy to calculate the design dependency between modules. While the MCS parameters are specific to certain physical interfaces and are therefore of limited use in program interdependency measurement, their method is noteworthy because it accounts for dependencies between components which are not directly connected. The electrical circuit analogy uses electrical resistance techniques to account for the effects of *all* modules in the system when calculating the design dependency between any two given modules. It accounts for more than just the modules under examination. This approach is desirable because a design change to one module may cause effects which propagate through intermediate modules to affect all modules in the

system, even those which do not share a direct interface with the module changed. This aspect of system dependency and effect propagation is directly applicable to program dependency measurement. Programs which are not directly connected to a program under evaluation may still have interdependency effects on that program. For example, Program A may depend upon Program B, which may depend upon Program C. While Program C is not directly connected to Program A, it can still be very important to program success. It can also cause severe adverse effects if it sufficiently disrupts Program B, the program which Program A depends upon directly. It is crucial to consider the effects of programs which may be distantly connected. In this work we refer to the magnitude of this distance separation as the *degree of interdependency*. This will be discussed further in Chapter 3.

Gaps in Current Understanding/Research

The existing research highlights the importance of interdependency measurement and illustrates possible applications of that measurement, including risk analysis and resource demand projections. However, none of the research thus far provides a model for quantitatively measuring unique and individual program interdependencies. While various interdependency-related parameters do exist, they do not objectively or sufficiently characterize program dependencies.

A maturity model can be used to provide the guidance to accurately and objectively measure program interdependency. This research presents a maturity model to measure program interdependencies based on the type of interdependency and the strength of each individual interdependency link. The maturity model provides the

structure to evaluate programs against a specific set of criteria in order to accurately determine and measure the types and strengths of program interdependencies.

Summary

At this point there is no accepted measurement model or method for quantitatively evaluating program interdependencies. While there has been significant research in the area of program dependency and interdependency, very little of this has focused on program interdependency measurement.

The maturity model serves as an ideal basis for a quantitative interdependency measurement method. Maturity models provide a way to quantitatively measure program qualities by using specified criteria to gauge the levels of those qualities. A maturity level or score is assigned based on the levels of the measured program qualities. Maturity models have already been adapted to measure program qualities such as interoperability.

While some research has been done on program interdependency measurement, this research is very limited. Current methods rely on either subjective judgments based on the perceived probability of program disruption, or confine interoperability measurement to the send-receive characteristics of program nodes or elements. Until this time, there has been no real way to quantitatively measure dependencies and interdependencies for an acquisitions program.

The next chapter introduces the interdependency measurement model and discusses its method of application to an acquisitions program.

III. Methodology

Chapter Overview

The purpose of this chapter is to present and explain the model used for program dependency and interdependency measurement. We first discuss the goals for the model in measuring program dependency and interdependency. Next we discuss the Interdependency Factors through which interdependencies are observed. The four Interdependency Factors are Funding, Technological, Support, and Systems Interaction Requirements interdependencies. These factors are the main means for identifying dependency and interdependency connections to other programs.

We also present the levels of program dependency and interdependency strength. Dependency links to other programs may vary in their strengths from program to program and may not even be the same strength when measured in both ways between two programs. That is, the strength of the dependency of Program A on Program B may not be the same as the dependency of Program B on Program A. Five levels of interdependency are presented in this thesis. These levels are used to measure the strength of the program dependency or interdependency relationship.

The measurement model is then presented with its method and structure for measuring program interdependencies in each of the four Interdependency Factors.

We then discuss Degrees of Interdependency, which allow us to evaluate interdependencies for programs not directly connected to the program of interest. For example, if Program A depends upon Program B, which depends upon program C, then

Program A is to some extent dependent upon Program C. Definition and measurement of this relationship are explained in this chapter.

Finally, the steps for proper application of the model are presented with guidelines for potential uses of the interdependency measurements.

Model Goals

The goal of the measurement model is to provide a method for accurately and quantitatively measuring program dependency and interdependency. In order to provide a complete measurement, the model is used to determine the types and strengths of dependencies and interdependencies present for a given program. The model does not simply indicate whether or not a program is dependent or interdependent; it also characterizes each of the program dependencies.

This information can then be used to assess program vulnerabilities incident to dependencies upon other programs. For example, if it is determined that Program A is heavily dependent upon Program B, then the program managers of Program A can take appropriate actions to provide mitigations in the case of an adverse effect to Program B.

The model can also be used to determine the criticality of a program to other programs. That is, the model can be used not only to determine the set of programs upon which Program A depends, but also the set of programs which depend upon Program A. This information can be used to show the importance of a program in terms of its effect on other acquisition programs. A key benefit of the interdependency measurement model is that it allows decision makers to determine the effects that one program may have upon other programs.

Interdependency Factors

Program dependencies and interdependencies manifest in several different types of ways. The types, or categories, of (inter)dependencies are called Interdependency Factors. These factors help to identify and characterize interdependency links between programs. Brown and Flowe have identified several of the ways in which program interdependency may manifest (Flowe, 2007; Brown, Kravchuk, & Owen, 2011). These include funding and resource interdependencies, among others. Other factors were identified based on evaluation of interdependency effects to past acquisition programs. These interdependency types were analyzed and combined to create the four basic interdependency factors presented in this research. These are: Funding, Technological, Support, and Systems Interaction Requirements. Understanding the program Interdependency Factors and their characteristics is critical in order to accurately measure program dependency and interdependency.

The Interdependency Factors may be grouped into two broader categories: Direct and Indirect factors. Direct factors are measured directly between two programs with no intermediaries between them. The programs are directly linked to each other. An example is a program whose system depends upon another program's system in order to function operationally. Direct factors may measure both dependencies and interdependencies. The Technological and Systems Interaction Requirements Interdependency Factors are Direct factors.

Indirect factors are measured between two programs which are connected by their ties to a common third party or entity. An example would be two programs which rely on

the same contractor or support office. While they may be largely unrelated, they are still both connected to a common organization, and so a level of interdependency may exist between them. The Interdependency Factors for Funding and Support are Indirect Factors. In order to measure these factors, we must first look at the support or funding entities to which the program is connected and then to any other programs connected to those entities. These programs are interdependent through their connection to those entities. This is illustrated below in Figure 2:

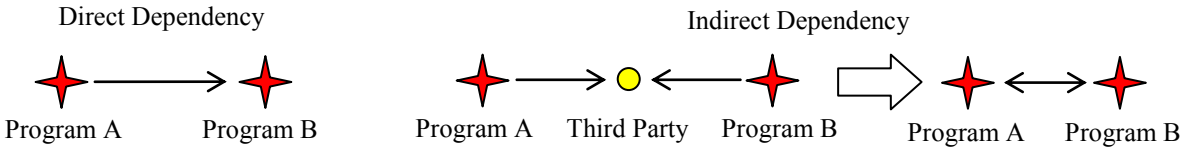


Figure 2. Direct dependency (left) and Indirect dependency (right)

Each of the Interdependency Factors may be evaluated by asking a specific question about the program. We will present each factor and its associated question in this section. Determining the answers to the questions will help identify program dependencies and interdependencies.

Funding Interdependency (Indirect)

Q: Where does the money come from and who else gets money from that source?

Funding Interdependency arises when multiple programs receive funding from the same source. For example, all programs funded by a given research office within the DoD are to some extent interdependent with each other.

The reason we are concerned with this interdependency is that disruptions in one program may affect other programs from a financial perspective. If a given program is adversely affected, there is often the potential that funds may be taken from other programs in order to mitigate effects to the first program.

To evaluate this Interdependency Factor, consider all funding sources or sponsors for the program. It is important to realize that a program may have multiple funding sources, especially if it is a large or joint program. Once all funding sources for the program have been identified, any other programs which receive funding from those sources should be also identified. The program is, at least to some degree, interdependent with all other programs which receive funding from any of its own funding sources. An illustration of funding interdependency is shown in Figure 3:

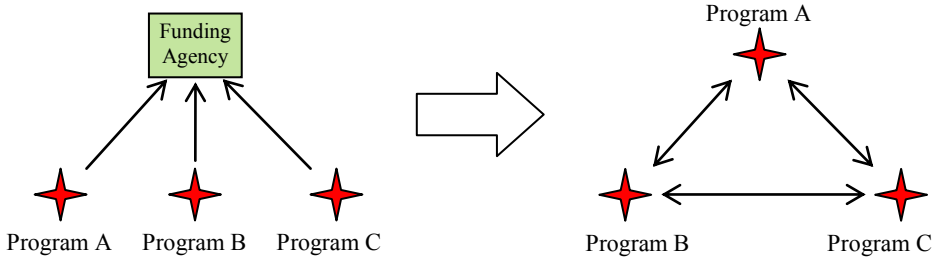


Figure 3. Funding Interdependency between three programs

It is may also be appropriate to evaluate higher levels of funding sources. Consider, What entities fund the program sponsors? What programs are funded by those higher-level entities? A program may also be interdependent with other programs which share funding sources at higher levels. In the example shown in Figure 4, Program A is

not only interdependent with Program B, but also with Programs C-F. The reason is that all programs share a funding source at a higher level.

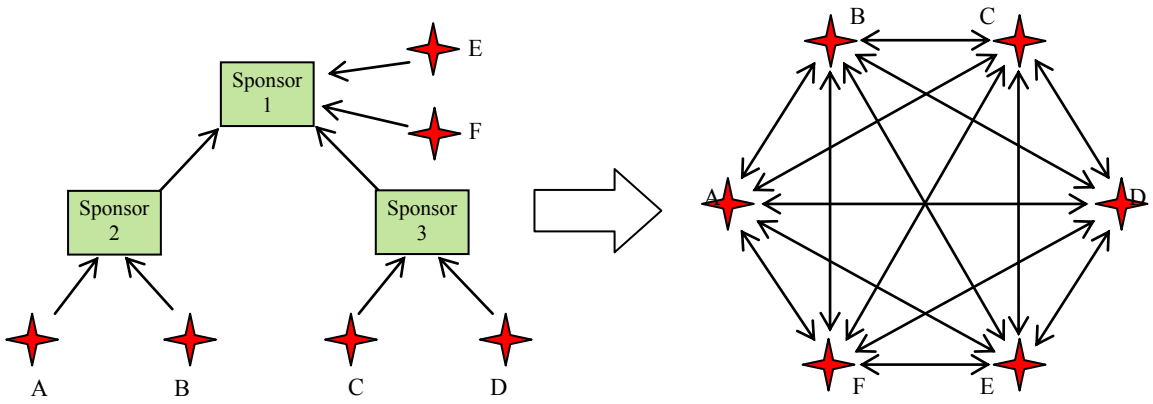


Figure 4. Diagram of interdependency based on higher levels of funding sources.

The strength of a funding interdependency depends upon several program considerations and is not necessarily the same strength both ways between programs. One program may be heavily dependent upon another program, while that program is only weakly dependent on the first program.

The first consideration is program priority. If, for example, Program A is seen as higher priority than Program B, then even if Program B suffers an adverse effect, Program A is less likely to be affected. Program A then, may be weakly dependent on Program B while Program B may be more strongly dependent on Program A. If programs are of approximately equal priority, then the levels of interdependency between them are more likely to be the same.

The second consideration is the level of funding interdependency. Programs which are connected at higher funding levels may be less interdependent than programs

which are connected at lower levels or programs which have the same immediate sponsor. Indeed, all acquisition programs within the Department of Defense ultimately share the same overall budget, yet the strength of program funding interdependency at such a high level may be very low, even negligible. The level at which funding interdependencies should be accounted for is up to the discretion of the program manager. It is important to account for higher-level funding interdependencies; however, dependency strength may decrease and become negligible as the level of the funding agency increases. Program funding interdependencies should be evaluated only to the highest level at which an adverse effect may be reasonably expected to propagate back to the program.

The third consideration is the budget and budget margin of the sponsor or sponsors. If the sponsors are themselves well-funded with budget margin, then adverse effects on one program are less likely to be spread to other programs funded by those sponsors. If the sponsor has very little or no budget margin, then adverse effects may be more likely to propagate between programs as the sponsor tries to find funds to cover expected shortfalls. These effects may vary with time as sponsors' budgets and commitments change.

Technological Interdependency – Technology, Processes, Materials, Data, etc (Direct)

Q: What programs are developing something new that is needed for our program?

Technological dependencies arise when a program requires an item, process, or technology which is being developed by a different program. Technological

dependencies are usually not two-way; i.e., Program A may require something that Program B is developing, but this does not mean that Program B necessarily requires anything from Program A. Thus, the Technological Factor usually concerns dependencies rather than interdependencies.

To evaluate the Technological Interdependency Factor, a functional or physical program decomposition is useful. Each physical and/or functional element of the system being developed by the program should be considered. The DoD Architecture Framework (DODAF) Systems and Technical views are good resources for examining system elements. We can examine program elements to determine if any of them are reliant on technology, processes, materials, data, etc., which have not yet been developed or which are still under development. If they are, then we can next determine which programs are developing those items. These programs are depended upon by the program under evaluation. Figure 5 illustrates technological dependency where an element of Program A depends upon an element of Program B. Program A is then dependent upon Program B.

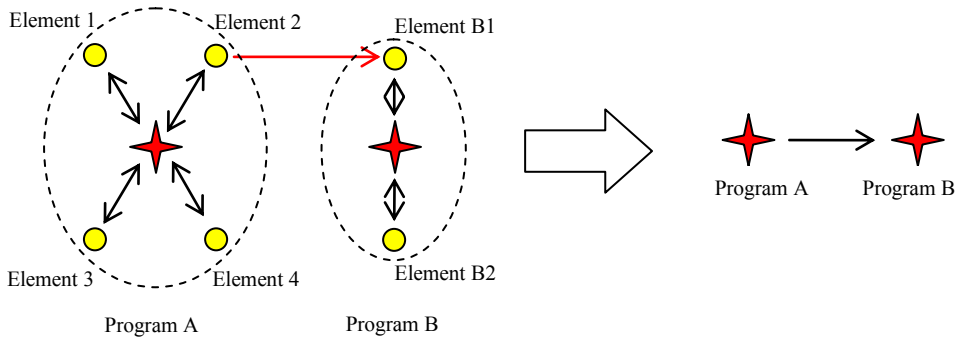


Figure 5. Technological dependency illustrated

Technology Readiness Levels (TRLs) can also indicate the potential presence of technological dependencies. Low TRLs for system components or elements may indicate a reliance on other programs which are maturing those technologies. High TRLs may indicate that system elements are already mature and do not depended upon developing technologies.

It is also important to consider new technologies, processes, etc., being developed by the program itself. This consideration is useful in determining the extent to which the program is depended upon by other programs.

The strength of the Technological Interdependency Factor is determined by the effect of in-development items upon program success. If another program is developing a new item which is only desired or which may be replaced by another item, then the program dependency is weaker. If a new technology is required for program success and there are no alternatives available, then the dependency is stronger.

Note that a program is not dependent upon itself if it is developing the technologies required. For example, if a satellite program is developing a new kind of sensor internally, then there is no technological dependency upon another program for the sensor. If, however, sensor development is external to the program, then there likely is a technological dependency.

Support Interdependency—Offices, Personnel, Contracts, etc. (Indirect)

Q: What other programs are supported by the agencies, organizations, contractors, or other entities responsible for supporting our program?

Programs may be interdependent if they are executed through the same agencies or organizations. For example, if two programs share a contractor and one of those programs suffers an adverse effect, the effect may spread to the other program as contractor personnel and resources are used to mitigate effects to the first program.

To determine Support Interdependencies, a decomposition of the program from a contractual or organizational standpoint is useful. The program office should consider all agencies or organizations with which they have a contract, support agreement, or Memorandum of Agreement/Understanding, and which will execute some part of the program. Considerations should also be given to subcontractors to these agencies which are responsible for large or critical program or system components. As these support entities are identified, the program office should identify other programs also supported by these groups. These programs may be interdependent with each other because of their link to a common support entity or organization as illustrated in Figure 6:

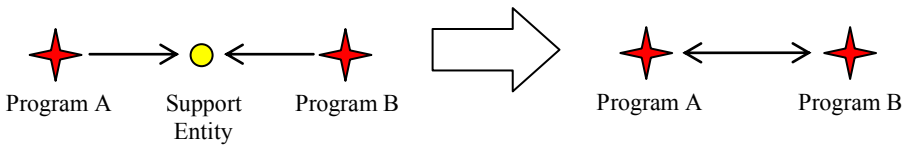


Figure 6. An illustration of support interdependency

The Support Interdependency Factor is two-way in that it identifies interdependencies between programs. Any program that depends upon another program from the Support perspective is also depended upon by that same program.

The strength of the Support Interdependency Factor depends upon the criticality of the program element supported by the support entity, the capabilities of the supporting entities, agencies, organizations, or contractors to support multiple programs or customers, and the priority of supported programs. If a given entity supports a minor or non-critical program element, as well as elements of other programs, then there is a weak interdependency between those programs. Likewise, if an entity has proven capable of simultaneously supporting multiple programs, or supports dissimilar aspects of multiple programs, then the interdependency strength may not be very strong. However, if critical program elements are supported by an organization which is also supporting other programs and/or has not proven capable of supporting multiple customers, then the interdependencies are much stronger. Additionally, program priority may influence the strength of interdependencies. A high priority program may be less dependent upon other programs from a support standpoint, while a relatively low priority program may be highly dependent.

Systems Interaction Requirements Interdependency (Direct)

Q: What in-development systems will our system depend upon operationally? What in-development systems will depend upon us?

From a programmatic standpoint, systems interaction requirements interdependencies arise when an acquisition program system depends upon another

system being developed by another program in order to function operationally. In order for a programmatic dependency or interdependency to exist, both systems must still be under development or not fully fielded. If one or both of the systems have been fielded, then the relationship is that of a *system* interdependency rather than a program interdependency.

To evaluate the Systems Interaction Requirements Interdependency Factor, we must examine how the system, once fielded, will relate to other systems. We must also consider where the requirement for the program and system originates. DODAF products or operational diagrams, such as the one shown in Figure 7, are again helpful in decomposing the system to find functional and operational links to other systems.

Figure 7. Operational connectivity diagram for the JSTARS system (http://www.defenseindustrydaily.com, 2005)

Systems which are linked operationally and which are still under development by an acquisition program have a programmatic dependency or interdependency. An example would be a new UAV system and a simulator for that system. Both may be developed concurrently under separate programs and both systems require the other in order to deliver operational capability. If the UAV program is adversely affected or cancelled, then the simulator program is in danger of being adversely affected as well.

Note that a *program* is generally not dependent upon a system that has already been fielded. We emphasize *program* here. For example, a new GPS-guided weapon *system* would be dependent upon the GPS constellation. However, because GPS is currently operational and functioning, the new weapon system *program* is probably not dependent upon the GPS program. However, if the weapons system interface with GPS depended upon an upgrade program being applied to the GPS fleet, then the program would be dependent upon the upgrade program.

While similar to the Technological Interdependency Factor, the Systems Interaction Requirements Interdependency Factor differs in that it considers external systems with which our system will interact operationally. The Technological Interdependency Factor focuses on developing items which will be used *internally* by our system, but which are currently under development by another program.

The strength of the Systems Interaction Requirements Interdependency depends upon the operational effects to the system under development if another depended-upon system program is adversely affected. If the system is still expected to be able to function operationally, or with only negligible degradation in capability, then the

program would be weakly dependent. If the other systems are critical to operational function, then the program would be strongly dependent upon the programs for those systems.

The Systems Interaction Requirements Interdependency Factor can be either one-way or two-way, depending on the specific case. That is, it can identify either dependencies or interdependencies. The criticality of each system to the other should be evaluated both ways in order to determine the dependency and interdependency strengths.

Interdependency Factors Summary

These four Interdependency Factors are sufficient to encompass the dependencies and interdependencies currently seen in DoD acquisition programs. These Interdependency Factors can be used to locate and identify program dependencies and interdependencies, both direct and indirect. Understanding these factors will allow the program manager to complete an accurate and comprehensive review of program dependencies and interdependencies.

Interdependency Levels

We define a program Interdependency Level as the strength of an interdependency link between two programs. Some interdependency links are stronger, or have more potential impact, than other links. An understanding of the strength of a given program's interdependency links is critical in measuring program interdependency. We use Interdependency Levels to gain this understanding. After a program's

interdependency links have been identified using the Interdependency Factors, the Interdependency Levels are used to determine the strength of those links.

The measurement model will use five Interdependency Levels. These levels have both numerical and descriptive values and range in value from Level 0 - Independent, to Level 4 - Mandatory. The levels are used to determine the strength of a dependency or interdependency between programs. A brief description of each level follows.

Level 0 – Independent

Programs have no dependency or interdependency connection as far as a certain Interdependency Factor is concerned. The programs do not have any influence over each other with respect to that factor.

Level 1 – Tangential

Programs have a largely insignificant bearing on each other with respect to an Interdependency Factor. With some adjustment, the program could continue to function even if other tangentially connected programs were removed.

Level 2 – Associated

The program is dependent or interdependent in ways that would result in the program being significantly affected if the associated program were disabled. The Program may lose some operational capability or experience some, probably recoverable, impacts to cost and schedule.

Level 3 – Dependent

The program is strongly connected to another program. The programs may share significant resources, support, or operational connections. The program would be severely impacted with respect to cost, schedule and/or performance if a depended-upon program were removed or adversely affected.

Level 4 – Mandatory

The program requires a connected program in order to function. The program may share critical resources with another program or depend upon the other for basic operational capability. The program cannot survive if the depended-upon program is removed or severely affected.

The Model

With the Interdependency Factors and Interdependency Levels as defined, we present the Measurement of Criticality and Interdependency Maturity Model (MCIMM).

Factor/Level of Dependency	Funding	Technological	Support	Systems Interaction Requirements
4 – Mandatory	Both programs are funded by the same office with the same budget with minimal margin	Technology or processes being developed by another program are Mandatory to program success	Programs share the same support personnel and critical elements share the same contracts	Program requires other program in order to function in an operational environment
3 – Dependent	Programs' funds come from the same well apportioned budget	Unavailability of technology or processes being developed by another program may have significant cost, schedule, or performance impacts	Major parts of the program share contracts or personnel with another program	Operational effects severely impacted by loss of other program.
2 – Associated	Portions of the programs' funds may come from the same office	Unavailability of technology or processes being developed by another program may have some impacts.	Parts of the program share contracts or support personnel with another program	Operational effects moderately impacted by loss of other program.
1 – Tangential	Limited portions of the programs' funds may come from the same agency	Technology or processes may be being developed concurrently by another program, or program has ability to substitute	Minor parts of the program may share contracts with parts of another program	Operational effects not significantly impacted by loss of other program.
0 – Independent	All funding for programs under consideration comes from different single sources	Program does not rely on any technology or processes being developed by another program	No personnel, offices, contracts, etc shared between programs	Able to achieve full operational effects without help from another program's system

The model uses the Interdependency Factors together with the Interdependency Levels to provide a framework for measuring dependency and interdependency between programs. Interdependencies are identified as the program is analyzed with respect to each Interdependency Factor. The strengths of the interdependencies are then quantified in accordance with the Interdependency Levels.

Degrees of Interdependency

Before we proceed, we need to discuss Degrees of Interdependency. We define Degree of Interdependency as the extent to which a program is separated or removed from another program which has interdependent effects upon it. For example, if Program A depends directly on Program B, then program A has a first-degree dependency on Program B. If Program B directly depends on Program C, but program A does not, Program A still has a second-degree dependency upon Program C. This relationship is illustrated in Figure 8:

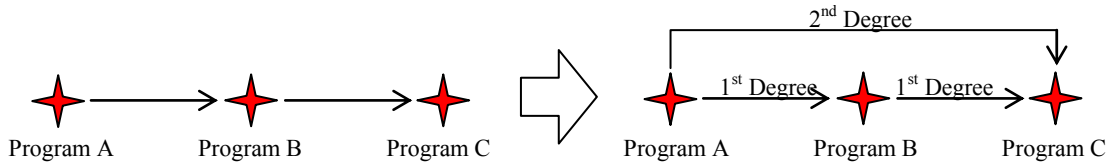


Figure 8. Illustration of Second-Degree dependency

Recognizing the various degrees of interdependency allows us to account for program dependencies that may otherwise be missed. The effects of higher-degree dependencies can propagate thorough programs to cause significant program effects through multiple degree of separation. For example, Program C in Figure 8, could cause

an effect to Program B. The disruption to Program B could then potentially cause an adverse program effect to Program A. It is important to understand that adverse effects may not originate solely from directly-connected, or first-degree, programs.

Eventually, nearly all programs within the DoD are linked by some degree of dependency. However, as the degree of dependency increases, the strength of the dependency generally decreases. For example, if Program A has a Level 1 - Tangential relationship with Program B, which has a Level 1 - Tangential relationship with Program C, then the relationship between A and C is negligible.

Quantifying the exact strength of higher-degree dependencies can be difficult. Each increasing degree of dependency adds another program between the primary program and the depended-upon program. These intermediate programs may act as buffers to absorb adverse program effects before they can propagate back to the primary program. In order to account for the change in dependency strength based on degree of dependency, the following method is proposed:

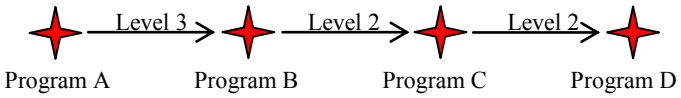
Let $S_{0,x}$ represent the dependency strength between the original program, Program 0, and Program x where x is an integer ≥ 2 and Program x is of x degree separation from Program 0. Let $S_{0,1}$ represent the strength of dependency between Programs 0 and a first-degree dependent program, Program 1, and let $S_{x\text{-}1,\,x}$ represent the strength between Programs x-1 and x. Note that x-1 ≥ 1. Then $S_{0,\,x}$ is given by the following equation:

$$S_{0,x} = 0.25^{x-1}\left(S_{0,1} * \cdots * S_{x-1,x}\right)$$

Equation 2. Higher-degree dependency strength measurement

This equation accounts for the decreasing strength of program dependency as the degree of dependency increases. It also allows critical higher-degree dependencies to be identified and their criticality maintained. This is illustrated in the following examples.

Example 1:

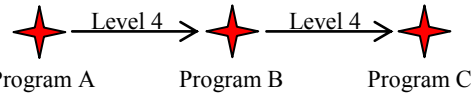


$$S_{A,D} = 0.25^{(2)}(S_{A,B} * S_{B,C} * S_{C,D})$$

$$= 0.0625(3*2*2)$$

$$= 0.75$$

Example 2:



$$S_{A,C} = 0.25^{(1)}(S_{A,B} * S_{B,C})$$

$$= 0.25(4*4)$$

$$= 4$$

In the first example, the importance of Program D falls off dramatically as far as dependency with Program A. However, in the second example, Program C is still critical to Program A with a Level 4-Mandatory dependency. The reason is that Programs A and B and Programs B and C have Level-4 Mandatory Dependencies with each other. If an adverse effect happens to Program C, it will certainly cause an adverse effect to Program B. This in turn will cause an adverse effect to Program A because of the Level-4 Mandatory dependency of Program A on Program B.

As mentioned, almost all programs within the DoD share some degree of interdependency through their connections to other programs. However, because the level of interdependency decreases with increasing degree, and because it may be unreasonable to track *all* of the higher degree dependencies, we suggest that any dependencies or interdependencies with an assessed level of less than one should be disregarded. We will use this approach for the remainder of this work.

Steps for Dependency and Interdependency Measurement

We now present the steps that should be followed when evaluating a program with the MCIMM model. The steps are illustrated in Figure 9:

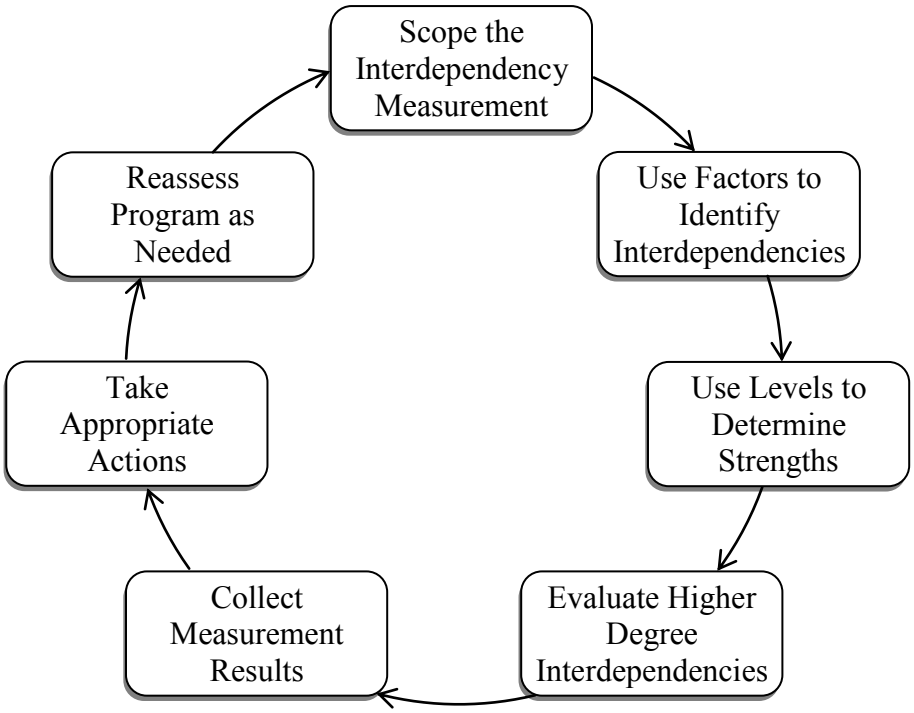


Figure 9. Steps for application of the MCIMM model

The scope of the measurement is the first thing considered when measuring program dependency and interdependency. The scope determines the maximum degree of interdependency we will measure. The appropriate scope for the measurement may vary depending on the size of the program being measured, the frequency of the measurement, the available program data, or the available resources to make the measurement. An office or agency may also wish to only measure program interdependencies for the programs within its control, or that agency may wish to evaluate interdependencies with respect to all other programs for which they can obtain data. In any case, the scope of the program interdependency measurement should be the first thing determined before proceeding.

The next step is to use the Interdependency Factors to analyze the program and identify dependencies and interdependencies with other programs. The program must be analyzed with respect to each factor separately. Direct and Indirect factors require different methods of evaluation, as discussed previously. Program decompositions may be helpful, especially in evaluating the Technological, Support, and Systems Interaction Requirements factors

Next, the Interdependency Levels should be used to measure the strength of each identified program dependency. The MCIMM presents the Levels for the different Interdependency Factors.

Second-degree and higher interdependencies can now be evaluated. This step depends on the scope of the measurement. The preceding two steps applying the Interdependency Factors and Levels are repeated for each depended-upon program

instead of the original program. The programs are evaluated for their own dependencies on other programs. Depending on the strength of those dependencies, the higher-degree programs may have significant dependency or interdependency effects on the original program.

The next step is to collect the results of the measurements. A tabulated list of interdependency links and strengths – such as the one shown in Table 1 – may be used. Separate tables may be used for second-degree or higher interdependencies.

Table 1. Example of a list of program interdependencies

Program Depended Upon	Type of Dependency	Degree	Dependency Strength
Program A	Funding	1st	4
Program B	Technological	2nd	3
Program C	Support	1st	2
Program D	Technological	1st	3
Program E	Requirements	2nd	2
Program F	Support	2nd	2.25

The results can yield several key parameters that can help us understand the dependency and interdependency characteristics of a program. The total number of dependencies can tell us how many other programs are depended upon by the program. The average strength of dependency, standard deviation of strength, maximum strength, and number occurrences of the maximum strength can give us an idea of how much a

program generally depends upon other programs (See Appendix A for calculations). Table 2 provides an example summary of interdependency metrics.

Table 2. Sample of interdependency metrics

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
Program X	23	2	2.15	0.38	4	2

These metrics give the program manager a complete picture of program dependency. While each individual dependency can be evaluated for its probability to cause adverse effects, the overall measurements provide an indication of how dependent the program is. The average strength of program interdependencies and other metrics may also be correlated to other program factors, such as cost or risks, in the same way that Brown and Flowe correlated the N_e factor with program resource demand (Flowe, et al., 2010). This correlation is beyond the scope of this research, however, as our focus is to establish the model and the interdependency metrics.

The next step is to take appropriate actions. Critically depended-upon programs can be identified through the interdependency measurement process. Program offices can then put mitigations in place in case these critical programs are adversely affected. They can also work to prevent the occurrence of adverse effects to critical programs. Program protection can be greatly enhanced by identifying program vulnerabilities through dependency and interdependency measurement and taking appropriate steps to reduce those vulnerabilities.

The final step is to re-assess the program as needed. The interdependency measurement provides a "snapshot-in-time" assessment of program interdependency. As programs move forward, as requirements evolve, and as program risks are retired, program dependencies and interdependencies change. Because of this, programs should be re-evaluated routinely in order to maintain an up-to-date and accurate measurement of program dependencies. This may entail annual, monthly, or quarterly measurements, or re-measurements based on significant program changes or events. The frequency of program interdependency measurement should be determined early in the life of the program.

Criticality Measurement

The MCIMM model can also be used to measure program criticality. We define program criticality as the strength of the dependencies of another program or programs upon the program of interest. This measurement will allow the program manager to determine the impacts that effects to the original program may have upon other programs.

To measure program criticality, the same basic steps are followed that are used in measuring program dependency. The difference is that instead of using the interdependency factors to find programs upon which the original program is dependent, we look for programs which will depend upon the program being measured. The method for evaluating dependent programs varies for each of the different Interdependency Factors.

For the Funding Interdependency Factor, the set of interdependent programs is usually the same. This is because the funding factor measures interdependencies rather

than dependencies. Any programs which Program A depends upon from the standpoint of the Funding Interdependency Factor also depend to some extent upon Program A. There may be differences in the strengths of the dependencies based on program priority or other program factors, but the programs are still interdependent. This is illustrated in the example in Chapter IV.

For the Technological Interdependency Factor, the set of dependent programs will likely be completely different. If Program A depends upon Program B for a developing technology, that does not necessarily mean that Program B is dependent upon Program A. When evaluating this factor for program criticality, we look for programs which will make use of technology, materials, data, or processes that are being developed by our program. These programs may be dependent upon our program. The strength of the dependencies can be measured using the Interdependency Levels established in the MCIMM.

Because the Support Interdependency Factor identifies program interdependencies rather than dependencies, the set of dependent programs and depended-upon programs is the same. There may be differences in dependency levels, again usually based on program priorities, but any program depended upon by another program from a Support perspective also depends upon that other program.

For the Systems Interaction Requirements Interdependency Factor, the sets of dependent and depended-upon programs may be similar or different. Dependent programs are those for systems which will depend upon our system operationally. They can be identified using DODAF products in the same way that depended-upon programs

are identified. However, the sets of dependent and depended-upon programs may not always be the same.

Once we have evaluated the Interdependency Factors for program criticality, we can proceed with the remainder of the steps for the MCIMM model. The measurements can be of the same form or type as those shown in Table 2, and will indicate how heavily depended-upon, or how critical, the program is to other programs. This can help illustrate the importance of protecting the program and the potential extended consequences of adverse program effects upon other programs. Decisions affecting the program can also be evaluated for their impacts to other programs. The MCIMM model can be a powerful tool for identifying and assessing the dependent and interdependent relationships of acquisition programs.

Summary

The MCIMM model allows us to quantify program interdependency strengths. The model can be used to identify the most important program dependencies, allowing the program manager to determine which programs have the potential for the most significant interdependency effects. The program manager can use this knowledge to help protect the program against those effects.

We have discussed program dependencies and interdependencies and how to measure them. We have discussed how program dependencies and interdependencies may be identified using the four Interdependency Factors and how their strength may be determined using the Interdependency Levels. We then presented the MCIMM model as a tool to identify and measure program dependencies and interdependencies. We also

discussed higher-degree dependencies, or dependencies upon programs not directly connected to the primary program, but instead connected to other depended-upon programs. We presented the steps for application of the MCIMM model to measure program interdependency and finally discussed the use of the model to measure program criticality.

The next chapter demonstrates application of the model to a space acquisition program. We will analyze the program with respect to the four Interdependency Factors, and measure dependency strengths using the Interdependency Levels of the MCIMM model. We will consider higher-degree interdependencies as well, as we apply the model. The program will be measured twice at different timeframes to demonstrate how program interdependencies may change with time. Finally, we will measure program criticality by using the model to identify dependent programs and to assess the strengths of those dependencies.

IV. Analysis and Results

Chapter Overview

The MCIMM model can be used to assess the dependency and interdependency characteristics of an acquisition program. These characteristics can highlight program vulnerabilities due to effects from other programs, as well as identify program criticality to other programs. In this chapter, we will apply the MCIMM to a space acquisition program in order to identify the dependencies and interdependencies of the program. We will consider first- and second-degree interdependencies in this measurement.

The program under evaluation in this example is an actual acquisition program that was recently completed. However, the program name and the names of other programs, items, and support agencies, have been masked. Nevertheless, the interdependency relationships and program characteristics are accurate and reflect the actual dependencies identified for the programs considered.

The program we will evaluate is the StarSat program, funded and administered by the Space and Satellite Development Office (SSDO). StarSat is a rapid-development program and, at program initiation, is scheduled to launch within three years. StarSat will carry a new type of payload, called SSP (StarSat Payload). This program will be evaluated for dependencies and interdependencies using the MCIMM model to apply the Interdependency Factors and Interdependency Levels. We will take measurements at program initiation and again at a time two years later.

Measurement Scope

For this example we will consider first- and second-degree dependencies. We will evaluate all programs which are depended-upon by the StarSat program and all programs which are depended upon by those programs. Because this is a demonstration of the method for application of the model, we will limit the scope of the measurement to Second-Degree interdependencies.

Now that we have determined the scope of the measurement, we are ready to use the Interdependency Factors to identify the program dependencies.

StarSat Funding Interdependencies

We will start with Funding Interdependency Factor. This is an indirect factor because we are not looking for dependencies upon program funding agencies, but rather we are looking for other programs funded by those agencies.

For the StarSat program, all funding comes from the Space and Satellite Development Office. At the start of the StarSat program, the SSDO was carrying two other programs which were dependent on its budget. These were the EagleSat and NovaSat programs. This is illustrated in Figure 10:

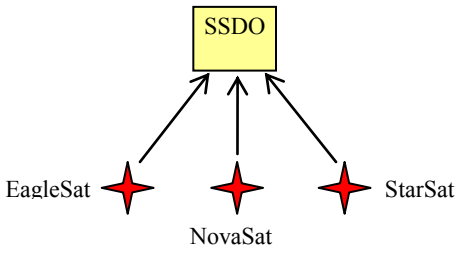


Figure 10. Programs funded by the SSDO at the start of the StarSat program.

Because each of these units depends upon the SSDO for support and because the SSDO budget is finite, effects on one program may spread to the other programs. If NovaSat were to experience a major overrun, the SSDO may decide to divert funds from the other programs in order to help NovaSat recover. This would likely cause schedule impacts to the other programs until funding can be recovered, and it may affect performance as well. Likewise, if EagleSat were to experience a major under-run, that may increase potential funding margins for the other programs. Because each program has the ability to affect the other programs funded by the SSDO, the three programs are interdependent. Figure 11 illustrates this funding interdependency:

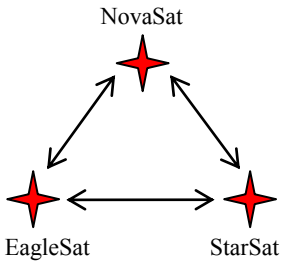


Figure 11. Interdependency between the EagleSat, NovaSat, and StarSat programs

We must also consider the agency that funds the SSDO. However, ORS is funded directly by the Office of the Secretary of Defense (OSD). Because of this, the SSDO programs are all interdependent to some degree upon every other DoD acquisition program in existence. However, because the DoD portfolio is so large and there are so many programs, the strengths of those interdependencies are estimated to be negligible. This illustrates the point that the scope of the measurement needs to be carefully considered. In some cases, such as large ACAT I programs, it may be appropriate to

consider funding dependencies upon other programs at the DoD level, or perhaps even higher. However, for the purposes of StarSat, the budget is contained at the SSDO level, so it is unlikely that overruns or underruns of a non-SSDO, DoD-level program would have any effect on StarSat.

At this point we are ready to evaluate the strength of the funding interdependencies between the three programs. The strength of the interdependency depends on the degree to which programs are funded from the same budgets, the funding and budget levels of the funding agencies, and the different priority levels of the programs. At program start, the SSDO itself is fully funded, and no funding difficulties are anticipated. Each program is entirely funded by the SSDO and is of approximately equal priority. Based on these factors, the interdependencies between all three programs would be Level 3 – Dependent. We then have the interdependencies shown in Table 3:

Table 3. Funding interdependency for StarSat at the start of the program

Program\Program depended upon	EagleSat	NovaSat	StarSat
EagleSat	-	3	3
NovaSat	3	-	3
StarSat	**3**	**3**	**-**

Because we are measuring the dependencies of the StarSat program, we count two Level 3 dependencies on the EagleSat and NovaSat programs. The other measurements may be used later to determine the criticality of the StarSat program to other programs.

It is important to note that the programs in this case are of the same priority. If this were not the case, then the interdependencies between the three programs may not be equal. We will illustrate funding interdependency differences based on priority later.

StarSat Technological Interdependencies

The Technological Factor considers developing program resources, including technologies, materials, processes, data, etc. This is a direct measurement because we look at other acquisitions programs directly rather than an intermediary agency or group.

In order to measure Technological dependencies, we can decompose the program physically, functionally, or both. A basic physical decomposition of the StarSat program is presented in Figure 12. Note that any program may be decomposed further as desired by the program manager.

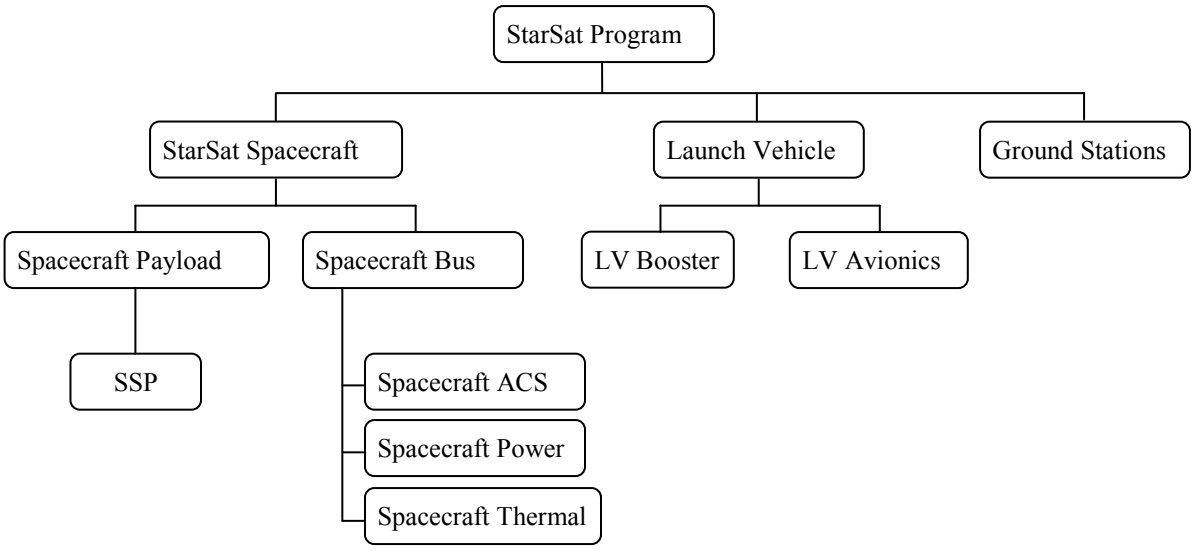


Figure 12. A basic decomposition of the StarSat program

Once we have decomposed the program to the appropriate level, we can analyze the components for Technological dependencies. The individual making the measurement determines the appropriate level for decomposition, but a full and rigorous decomposition is encouraged. This will prevent the omission of possibly critical technological dependencies at lower component levels.

In the case of StarSat, analysis of the program identifies four developing technology items. The first is the SSP payload. The second is the spacecraft power system, which will use a developing type of solar panel. The third item is the launch vehicle. The StarSat program selected the FireBird Launch Vehicle (LV), which is being developed under another program. Finally, the spacecraft is being designed and integrated under a new process unique to StarSat. The process allows for rapid fielding of the system, but has never been tried in space vehicle acquisition.

Of the four technological items identified, two of them are being developed internally by the StarSat program. These are the SSP payload and the spacecraft design process. The other two, the solar panels and the FireBird LV, are new items being developed by other programs. Because StarSat is planning to incorporate these technologies, it is dependent upon these other programs. The launch vehicle is developed under the BigSat satellite program, and the solar panels are developed under a separate technology initiative. These dependencies are represented in Figure 13. Note that the diagram is simplified and shows only those elements involving technological dependencies.

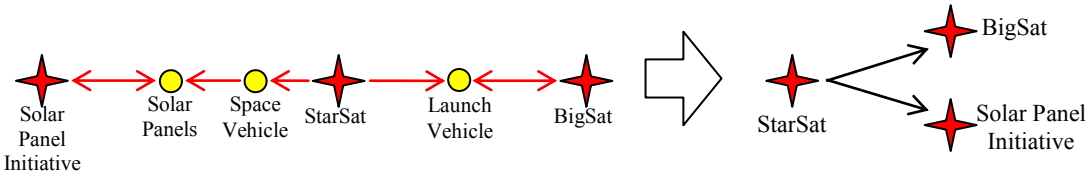


Figure 13. StarSat technological dependencies

It is important to remember that these technological dependencies are not necessarily two-way. In this case, the StarSat program is dependent upon the two other programs developing technologies but not vice versa.

We can now determine the strengths of the Technological dependencies. For the solar panels, the new technology would benefit the program; however, proven, existing solar-panel technologies could be substituted if there are problems with the development of the new panels. There would possibly be some cost or schedule impacts to the program but not to the extent that the program would be endangered or severely affected. This would be a Level 2 – Associated dependency.

The launch vehicle is a different matter. The program office determined that the FireBird LV was the only suitable launch vehicle for the mission. It could reach the desired orbit, and estimated cost is within the program budget. The next cheapest launch vehicle that could meet mission requirements would cost almost three times as much. Because the FireBird is required for the StarSat mission, and because the vehicle is being developed by the BigSat program, StarSat has a Level 4 – Mandatory dependency on the BigSat program. Table 4 summarizes the external technological dependencies for the StarSat program:

Table 4. StarSat Technological Dependencies

Program\Program depended upon	BIGSAT	Solar Panel Initiative
StarSat	**4**	**2**

StarSat Support Interdependencies

The next Interdependency Factor is the Support Factor. This is an indirect factor. We will look for programs supported by the agencies, contractors, or other entities that support StarSat. In order to measure this factor, we will first decompose the program organizationally. We will look for any organizations or groups with which StarSat has a contract or Memorandum of Agreement or Understanding, or from which services are required in order to complete the program. We will also need to identify the entities that those groups rely on for support. We then look for other programs supported by those organizations. An organizational decomposition of the StarSat program is shown in Figure 14:

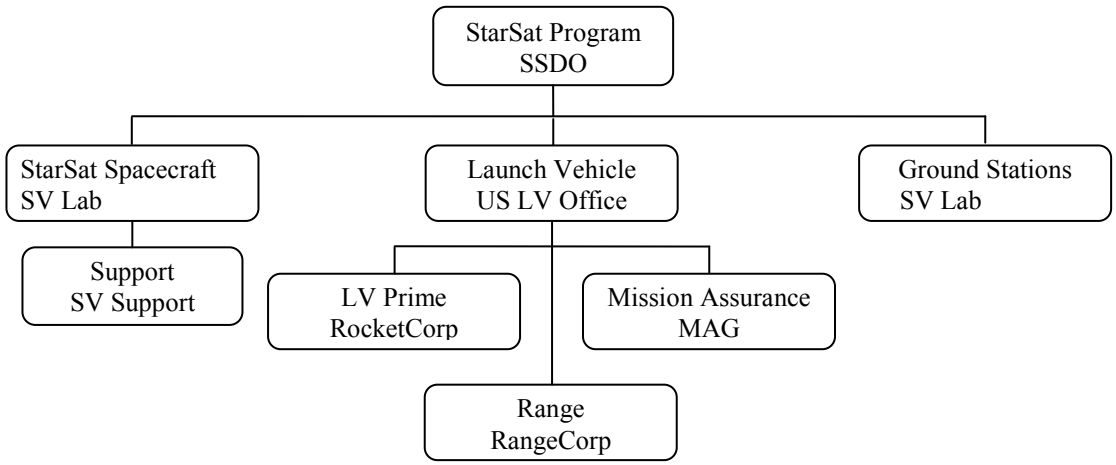


Figure 14. Organizational decomposition of the StarSat program (names changed)

The next step is to determine what other programs are supported by the support organizations and entities that support StarSat. The StarSat program may be dependent upon these programs based on the nature of support to each program. Support agency limitations in terms of supplies, schedule, capability, and manning should be considered when making this analysis.

For StarSat, each support agency was evaluated in terms of support commitments to other customers and dedication of resources to the StarSat program. A summary of the commitments of the support entities is shown in the Table 5:

Table 5. Summary of StarSat support entities

SV Lab	Personnel support two other programs. While StarSat is a priority right now, dependencies are expected with other SV Lab program efforts.
SV Support	Dedicated support personnel provided to SV team.
US LV Office	Supports multiple satellite programs with Launch Vehicle and Launch Services. Uses the same contracts and contractors to support multiple programs. Other programs include QuickSat, BigSat, LittleSat, NovaSat, EagleSat, and SafeSat. Dependencies expected here.
RocketCorp	Prime Contractor for Launch Vehicle. Provides LVs to US LV Office for use on multiple programs.
Mission Assurance Group (MAG)	Dedicated support personnel provided to LV team.
RangeCorp	Supports multiple programs providing launch range services to different customers. Current customer is RLD, though US LV Office programs will start to require support very soon. Interdependencies expected here with RLD programs.

The Space Vehicle (SV) Lab currently supports two other the programs in addition to StarSat. Some of the same personnel are used to support these programs. Their support contractor, SV Support, provides dedicated personnel and so no significant interdependencies are expected with respect to SV Support. The US LV Office and RocketCorp support multiple programs as well. Many of the same resources are used to support these programs. Finally, RangeCorp has another customer, the Rocket Launch Group (RLG), with a current program.

Figure 15 shows a visualization of the support entities and support interdependencies for StarSat:

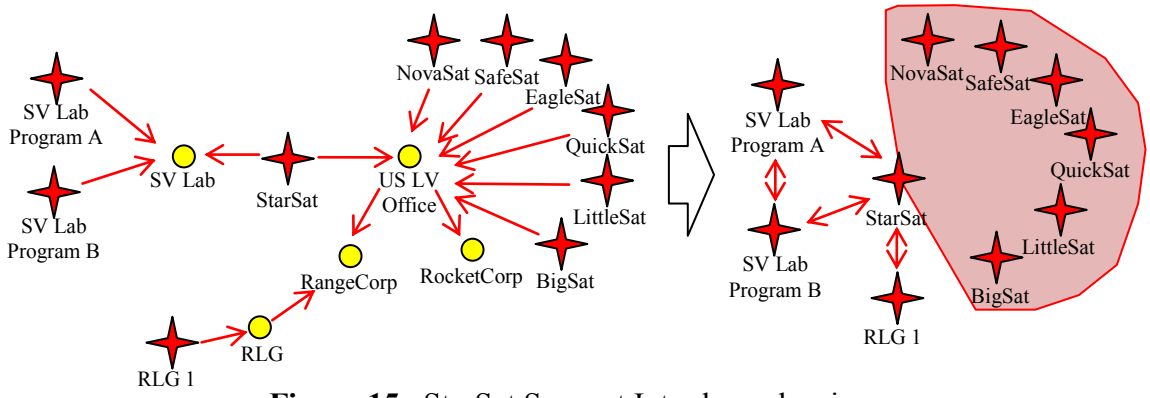


Figure 15. StarSat Support Interdependencies

Note that all three SV Lab programs are interdependent with each other, as are all of the programs supported by the LV Office. StarSat and the RLG-1 program are also interdependent with each other because of their use of the same range.

With these interdependencies identified, we can now evaluate the strength of each interdependency. Table 6 shows the support interdependencies for StarSat:

Table 6. StarSat Support Interdependency strengths

Program	StarSat Dependency Strength	Dependencies upon StarSat
BigSat	4	2
NovaSat	3	3
EagleSat	3	3
SV Lab A	3	3
SV Lab B	3	3
SafeSat	3	3
QuickSat	3	3
LittleSat	3	3
RLG 1	2	2

With the exception of BigSat, each program supported by the US LV Office shares a Level 3 – Dependent interdependency. The reason is that a major part of each program, the launch vehicle, is supported by the same office and the same contractor: RocketCorp. BigSat, however, is the highest priority program and so is less dependent upon the other programs. This also means that the other programs are more dependent upon BigSat.

The SV Lab programs and the StarSat program also share a Level 3 – Dependent interdependency because the SV Lab is responsible for major portions of each program: the SV for StarSat and other systems for the other programs.

The RLG-1 program and the StarSat program share a Level 2 – Associated interdependency. The reason is that while they both use the same range, they are administered by different support agencies and contracts. RangeCorp is a contractor to

the LV Office for StarSat and a contractor to RLG for the RLG-1 program on a separate contract. The LV Office is not part of the RLG program.

StarSat Systems Interaction Requirements Interdependencies

To evaluate this factor, we will decompose the StarSat program operationally (see Figure 16). The major questions are: Who, or what systems, will StarSat interact with in the field? What is driving the need for StarSat? What systems or equipment will users need in order to use StarSat? Are any of these systems currently in development?

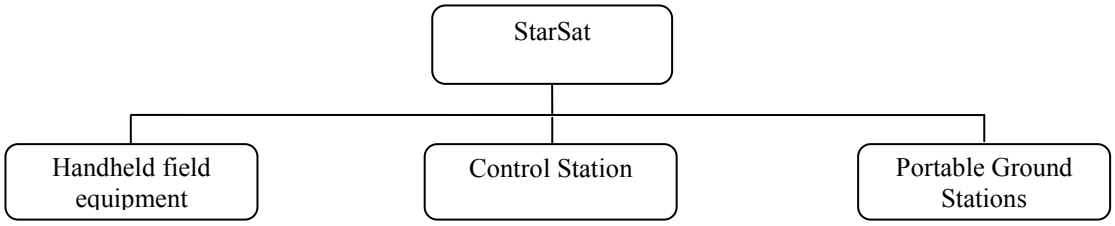


Figure 16. StarSat operational interactions

StarSat interacts with several different types of field equipment, a dedicated control station, and several portable ground stations. In the case of StarSat, the field equipment with which StarSat will interface already exists and is already in service. Therefore, even though there is a *systematic* interdependency with this equipment, there is no *programmatic* interdependency because the equipment is not part of a current acquisitions program in the development or procurement phase. The ground and control stations are being developed internally by the StarSat program, so there are no external programmatic dependencies for these items either.

For the StarSat program, we do not identify any *programmatic* systems interaction requirements dependencies. There are certainly operational dependencies and interdependencies at the systems and component levels; however, there are no dependencies identified based on programs for systems that are under development or that have not yet been fielded.

Measurements

Now that we have used the Interdependency Factors to identify program dependencies and interdependencies and the Interdependency Levels to measure program dependencies, we are ready to integrate the first-degree dependency measurements. Table 7 summarizes the measured program dependencies and interdependencies for all four factors for the StarSat program:

Table 7. StarSat Dependency links and strengths at start of program

Factor\Strength	4-Mandatory	3-Dependent	2-Associated	1-Tangential	0-Independent
Funding	0	2	0	0	Not Measured
Technological	1	0	1	0	Not Measured
Support	1	7	1	0	Not Measured
Requirements	0	0	0	0	Not Measured
Totals	2	9	2	0	-

We have identified 13 dependencies for the StarSat program. Of these, the Technological and Support dependencies on the BigSat program for the FireBird LV and LV Office support are Level 4 – Mandatory dependencies. Nine more are Level 3 –

Dependent and are funding- and support-related. There are two Level 2 – Associated, dependencies that are related to Technological and Support factors. No Level 1 – Tangential or Level 0 – Independent dependencies were observed. Figure 17 shows the full network of identified program dependencies for the StarSat program:

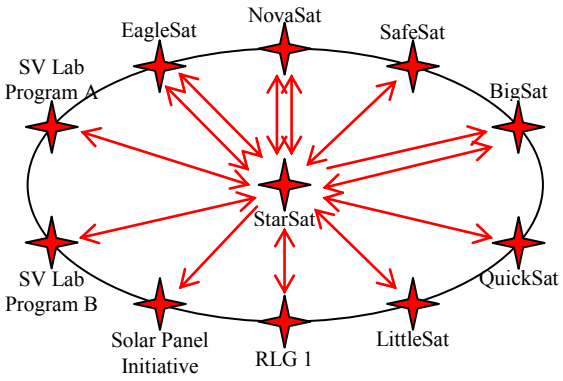


Figure 17. StarSat program dependency links at program start

Note, however, that the BigSat, NovaSat and EagleSat programs are all depended upon in more than one factor. StarSat is dependent on BigSat in the technology and support factors. StarSat is interdependent with both EagleSat and NovaSat for funding and support. Depending on the situation, this multiplicity of interdependencies could create a complication. If we wish to know the total number of dependencies, then the data in Table 7 is appropriate. The program manager has a complete view of how the program relates to other programs and the ways in which the program may be affected by other programs. The program manager can plan mitigations for every program dependency accordingly. However, if we are interested in knowing the number of *programs* upon which StarSat is dependent, then the total number of dependency *links* may be misleading. For example, if we had dependencies upon NovaSat in all four of the

Interdependency Factors but did not depend upon any other programs, then we would say that we are dependent on only one program: NovaSat. Even though there are four dependencies, one for each factor, we are only dependent on one program. Accounting for all interdependency links to the program may lead to undue emphasis of the importance of that program or may make the original program being measured appear more dependent than it actually is. In cases where we have multiple links to a single program we may wish to take the strongest links to that program and disregard the weaker links.

The proper approach depends upon the situation. For vulnerability assessment and protection, it may be best to account for all interdependency links. This approach avoids the loss of dependency information and the potential failure to recognize program vulnerabilities. To measure overall program dependency characteristics, however, it may be more appropriate to account for only one unique program dependency per program. This thesis will take the second approach for the remainder of this example because we are demonstrating overall dependency measurement for the StarSat program. Table 7 is then modified as shown in Table 8 below:

Table 8. StarSat program dependencies

Factor\Strength	4-Mandatory	3-Dependent	2-Associated	1-Tangential	0-Independent
Funding	0	2	0	0	Not Measured
Technological	1	0	1	0	Not Measured
Support	0	5	1	0	Not Measured
Requirements	0	0	0	0	Not Measured
Totals	1	7	2	0	-

For BigSat, we determine that technological interdependency is stronger than the support interdependency, so the support interdependency is disregarded. This is up to the discretion of the program manager as the two links for BigSat are both Level 4 – Mandatory. The interdependencies for EagleSat and NovaSat are both Level 3 – Dependent for funding and support. The choice of which to disregard is again left to the program manager. In this case, let us say that the program manager believes that disruptions due to agency funding are somewhat more likely than disruptions due to a common support agency. Then the support interdependency links are disregarded. Table 9 lists programs with first-degree dependency connections to StarSat.

At this point the number of links in the table matches the number of programs upon which StarSat is dependent. This method of assessing dependency and interdependency links is useful if we wish to determine the number of *programs* depended upon rather than the total number of interdependency links. However, the program office should keep in mind that multiple links to a given program may exist with additional dependencies and vulnerabilities.

Second-Degree Interdependencies

We have evaluated the programs upon which StarSat is directly dependent or interdependent. However, this initial group of programs may not be the only programs which could affect StarSat. Each of the directly connected programs may depend upon other second-degree programs. Those second-degree programs may in turn depend upon third-degree programs. If any of these higher-degree programs are adversely affected, those effects could propagate back to StarSat.

We will only measure up to second-degree dependencies in this example. To account for second-degree programs, we run the dependency model for each of the first-degree, or directly connected, programs. Table 9 summarizes the results for the StarSat program showing second-degree depended-upon programs for each of the directly first-degree programs (see Appendix B for calculations and data). Program names have again been masked for the Second-degree programs.

Table 9. List of StarSat second-degree dependencies

First-Degree Program	Second-Degree Program	Dependency strength (for First-Degree Program)
BigSat	BigSat Funding 1	2
	BigSat Funding 2	2
LittleSat	LittleSat Funding 1	3
	LittleSat Funding 2	2
	LittleSat Requirements 1	2
	LittleSat Requirements 2	2
	LittleSat Requirements 3	3
QuickSat	QuickSat Funding 1	3
	QuickSat Requirements 1	4
	QuickSat Requirements 2	4
SafeSat	SafeSat Funding 1	2
	SafeSat Technological 1	4
RLG-1	RLG Funding 1	3
	RLG Funding 2	3
	RLG Requirements 1	4
	RLG Requirements 2	3

We can then use Equation 2 to find the level of dependency for StarSat on each of the second-degree programs. The results are shown in Table 10. (See Appendix B for calculations.)

Table 10. Summary of StarSat second-degree dependencies

Program	StarSat Dependency Strength
BigSat Funding 1	2
BigSat Funding 2	2
LittleSat Funding 1	2.25
LittleSat Funding 2	1.5
LittleSat Requirements 1	1.5
LittleSat Requirements 2	1.5
LittleSat Requirements 3	2.25
QuickSat Funding 1	2.25
QuickSat Requirements 1	3
QuickSat Requirements 2	3
SafeSat Funding 1	1.5
SafeSat Technological 1	3
RLG Funding 1	1.5
RLG Funding 2	1.5
RLG Requirements 1	2
RLG Requirements 2	1.5

When we add these to the ten first-degree dependencies previously evaluated, the overall dependency measurement for the StarSat program changes dramatically. We now have 26 dependencies, ranging in strength from 1.5 to 4. These are shown in Table 11:

Table 11. StarSat total program dependencies; 1st and 2nd degree

Program	Degree	Dependency Strength	Dependency Factor(s)
BigSat	1st	4	Tech, Support
NovaSat	1st	3	Funding, Support
EagleSat	1st	3	Funding, Support
SV Lab A	1st	3	Support
SV Lab B	1st	3	Support
SafeSat	1st	3	Support
QuickSat	1st	3	Support
LittleSat	1st	3	Support
QuickSat Requirements 1	2nd	3	Requirements
QuickSat Requirements 2	2nd	3	Requirements
SafeSat Technological 1	2nd	3	Tech
RLG 1	1st	2	Support
LittleSat Funding 1	2nd	2.25	Funding
LittleSat Requirements 3	2nd	2.25	Requirements
QuickSat Funding 1	2nd	2.25	Funding
Solar Panel Initiative	1st	2	Tech
BigSat Funding 1	2nd	2	Funding
BigSat Funding 2	2nd	2	Funding
RLG Requirements 1	2nd	2	Requirements
LittleSat Funding 2	2nd	1.5	Funding
LittleSat Requirements 1	2nd	1.5	Requirements
LittleSat Requirements 2	2nd	1.5	Requirements
SafeSat Funding 1	2nd	1.5	Funding
RLG Funding 1	2nd	1.5	Funding
RLG Funding 2	2nd	1.5	Funding
RLG Requirements 2	2nd	1.5	Requirements

Depending on the scope of the measurement, it may be appropriate to evaluate third-degree and even higher interdependencies. These higher-degree dependencies are evaluated in the same way that second-degree interdependencies are evaluated. These higher-degree relationships have in the past had significant effects upon programs. High-degree interdependencies, especially when they are high-level interdependencies, can propagate effects through multiple degrees to affect a program, in some cases even causing catastrophic effects. For the purposes of this example, we have limited measurement to first- and second-degree interdependencies. However, the importance of searching out and measuring higher-degree interdependencies cannot be overstated.

Dependency Strengths

Based on Table 8, we can calculate the first-degree dependency metrics for StarSat. The average program dependency strength is 2.9, the Standard Deviation of the dependency strengths is 0.539, and the single maximum dependency strength is Level 4 – Mandatory (see Appendix B for calculations). The first-degree dependency measurements can be summarized as shown in Table 12:

Table 12. StarSat first-degree dependency summary

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	10	1	2.9	0.539	4	1

We can also calculate metrics for second-degree program dependency strengths as well as total program dependency strengths. The average strength is 2.016 with a standard deviation of 0.56 for second-degree program interdependencies (see Table 13), and an average strength of 2.36 with standard deviation 0.7 for all measured program interdependencies (see Table 14).

Table 13. StarSat second-degree dependency summary

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	16	2nd only	2.016	0.56	3	3

Table 14. StarSat total program dependency summary at program start

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	26	2	2.36	0.7	4	1

These measurements show how strongly the StarSat program depends upon other programs. Table 11 can also help determine the most heavily depended-upon programs, which can help program managers find the best ways to protect against adverse effects.

Take Action

The program office should be aware of developments, progress, or obstacles within the other programs which StarSat is dependent upon. From Table 11 we can see the most critical programs to StarSat. These are the programs to which StarSat is the most vulnerable for interdependency effects. The most critical dependency is on the BigSat program. Adverse effects to BigSat, which is the lead program for development of the LV, could have high potential to disrupt StarSat. The same is true for any second-degree programs which BigSat depends upon. If these second-degree programs are disrupted, the effects could spread through BigSat to StarSat. The program office should carefully monitor the status of the BigSat program in order to prepare for and prevent adverse interdependency effects. Additionally any of the programs which share support offices, particularly the LV Office and the SV Lab, should be monitored.

The program office can work to implement mitigation strategies in case of adverse effects in these areas. These strategies may be technical or contractual in nature, or may involve changes to concepts of operations. Once program dependencies have been identified and measured, the program office can best determine how to protect against adverse effects from depended-upon programs.

Continuing Measurements

The program office should continue to evaluate program interdependency for StarSat throughout the life of the program. While the model provides an accurate snap-shot-in-time assessment of program dependencies and interdependencies, these interdependencies will change as programs progress or are completed, and as new

programs emerge. At the conclusion of a program, the program interdependencies will likely be very different from the interdependencies at the inception of the program. By repeating the dependency and interdependency measurement process for StarSat as programs evolve, the program office will maintain awareness of changing program dependencies and changing vulnerabilities based on those dependencies. The program office should determine how often to conduct interdependency measurements in order to maintain an accurate and current assessment of program interdependency.

StarSat Interdependency Reassessment Two Years Later

We will briefly illustrate continuing measurements by applying the MCIMM model to StarSat two years after program start. We will focus on the total program dependency measurement.

Funding

The Funding Interdependency Factor can change greatly as programs are completed or as new programs emerge. This is the case with StarSat. Two years after program start, the EagleSat and NovaSat programs have been completed. These programs no longer influence StarSat. However, the SSDO is now funding two new programs: UltraSat and NewSat. These two new programs draw funds from the same source as StarSat and are therefore interdependent with the StarSat program. Figure 18 shows the updated funding interdependencies for StarSat:

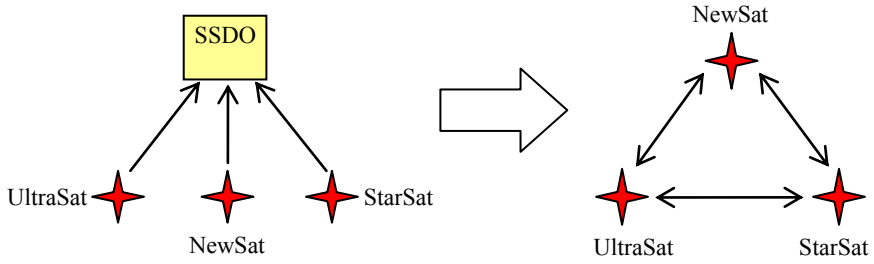


Figure 18. StarSat Funding Interdependencies at two years after program start

However, the UltraSat program is very high priority. If any adverse events occurred prior to the completion of UltraSat, funds would be pulled from the other two programs in order to keep the UltraSat going and to avoid a delay.

Conversely, the NewSat program is lower priority and is very far from being fielded. If there were an adverse effect to NewSat, the program would have time to recover and would likely not require additional funds, especially from higher priority programs.

This program priority characteristic effectively means that StarSat sees the UltraSat and NewSat missions at different interdependency levels. UltraSat would be at Level 4 – Mandatory, while NewSat would be at Level 2 – Associated. UltraSat would see StarSat as Level 2 – Associated, and NewSat as Level 1 – Tangential. That is, UltraSat funds would never be taken to compensate for an adverse effect to the NewSat program. NewSat would see UltraSat as Level 4 – Mandatory and StarSat as Level 3 – Dependent. These interdependencies are summarized in Table 15:

Table 15. Funding interdependency for StarSat two years after program start

Program\Program depended upon	UltraSat	NewSat	StarSat
UltraSat	-	1	2
NewSat	4	-	3
StarSat	**4**	**2**	**-**

This situation is illustrative of the differences that program priority can have on program interdependencies and thus shows why it is important to consider program priority when measuring interdependency. It also illustrates that funding interdependencies are not always the same strength both ways between two programs.

Technological

Technological dependencies can change as technologies develop or even as technological development efforts fail. Once a new technology or process has been fielded or successfully proven by another program, that dependency may be able to be reduced or retired.

It is also important to determine if there are any new programs that will use the technologies that StarSat itself is developing. These programs could add to or change the ways in which our program is depended upon.

In the case of StarSat, one of the major technological dependencies, the solar panel program, has been retired. The LV development under BigSat is the only remaining technological dependency as shown in Figure 19:

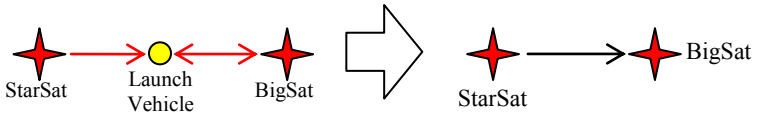


Figure 19. StarSat technological dependencies at two years after program start

The strength of the dependency is unchanged because the LV is still critical to the StarSat program. The new state of the technological dependencies for StarSat is shown in Table 16:

Table 16. Technological interdependency for StarSat two years after program start

Program\Program depended upon	BigSat
StarSat	**4**

Support

The Support Interdependency Factor changes as agencies and contractors take on new obligations and fulfill old ones. Two years after the start of StarSat, the LV Office and RocketCorp have completed support for EagleSat and NovaSat but have started support for three new programs: the BrightSat program, UltraSat, and FarSat. RangeCorp has completed support for the RLG-1 program, but has also started support for LittleSat. Additionally, the SafeSat program has decreased in priority. There are no changes to support from other entities. Figure 20 shows the updated interdependencies for the Support Interdependency Factor for StarSat.

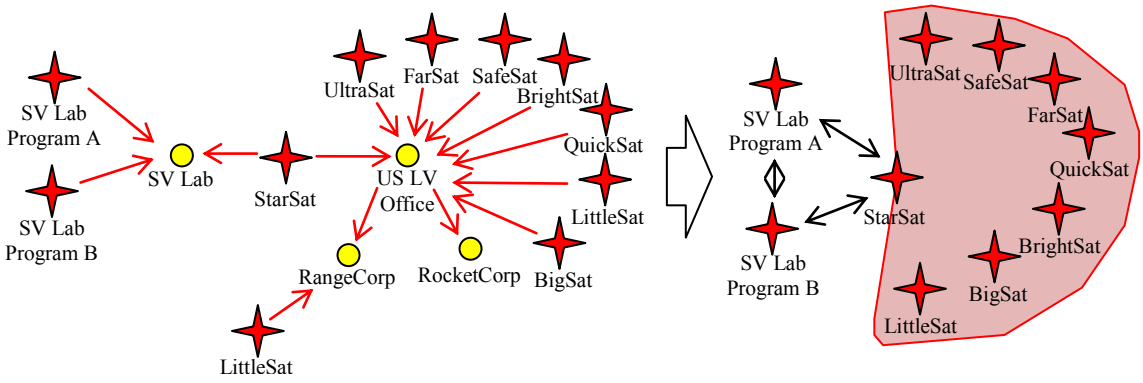


Figure 20. StarSat support interdependencies at two years after program start

Table 17 shows the updated support interdependency strengths for StarSat two years after the start of the program. Notice that although the StarSat dependency on SafeSat has decreased, SafeSat still has a Level 3 dependency on StarSat.

Table 17. Support Interdependencies for StarSat two years after program start

Program	StarSat Dependency Strength	Dependencies upon StarSat
BigSat	4	2
SV Lab A	3	3
SV Lab B	3	3
UltraSat	3	3
FarSat	3	3
BrightSat	3	3
QuickSat	3	3
LittleSat	3	3
SafeSat	2	3

Systems Interaction Requirements

As requirements evolve and operational concepts change, systems interaction requirements interdependencies may change along with them. In the case of StarSat however, all interdependent systems have already been fielded and there have been no changes to concepts of operations or requirements. While new programs may eventually emerge which will be depend upon StarSat operationally, there are no changes to systems interaction requirements interdependencies at this time.

At this point, the first-degree dependencies for StarSat can be represented by Table 18. For UltraSat and BigSat, we again count only the unique interdependent programs rather than the total number of interdependency links.

Table 18. StarSat program dependences two years after program start

Factor\Strength	4-Mandatory	3-Dependent	2-Associated	1-Tangential	0-Independent
Funding	1	0	2	0	Not Measured
Technological	1	0	0	0	Not Measured
Support	0	6	0	0	Not Measured
Requirements	0	0	0	0	Not Measured
Totals	2	6	2	0	-

Second-Degree Dependencies

We would next evaluate higher-degree dependencies for StarSat. An updated analysis would show the following second-degree interdependencies, as shown in Table 19 (see Appendix B for calculations):

Table 19. StarSat second-degree dependencies two years after program start

First-Degree Program	Second-Degree Program	Dependency strength (for First-Degree Program)
BigSat	BigSat Funding 1	2
	BigSat Funding 2	2
LittleSat	LittleSat Funding 1	3
	LittleSat Funding 2	2
	LittleSat Requirements 1	2
	LittleSat Requirements 2	2
	LittleSat Requirements 3	3
QuickSat	QuickSat Funding 1	3
	QuickSat Requirements 1	4
	QuickSat Requirements 2	4
BrightSat	BrightSat Funding 1	3
	BrightSat Funding 2	3
FarSat	FarSat Funding 1	3
SafeSat	SafeSat Funding 1	2
	SafeSat Technological 1	4

Measurements

Based on the new interdependency analysis, we can update the interdependency measurements for StarSat. We still have 10 first-degree program dependencies, although they are slightly different now, and we have 15 second-degree dependencies for a total of 25. The summary table of all program dependencies and their strengths is given in Table 20. (See Appendix B for the calculations.) This table provides on update of program vulnerabilities. Note that we again only evaluate the number of programs upon which StarSat is dependent or interdependent, not the total number of interdependency links.

Table 20. StarSat program dependencies two years after program start

Program	Degree	Dependency Strength	Dependency Factor(s)
BigSat	1st	4	Tech, Support
UltraSat	1st	4	Funding, Support
SV Lab A	1st	3	Support
SV Lab B	1st	3	Support
QuickSat	1st	3	Support
LittleSat	1st	3	Support
FarSat	1st	3	Support
BrightSat	1st	3	Support
QuickSat Requirements 1	2nd	3	Requirements
QuickSat Requirements 2	2nd	3	Requirements
LittleSat Funding 1	2nd	2.25	Funding
LittleSat Requirements 3	2nd	2.25	Requirements
QuickSat Funding 1	2nd	2.25	Funding
BrightSat Funding 1	2nd	2.25	Funding
BrightSat Funding 2	2nd	2.25	Funding
FarSat Funding 1	2nd	2.25	Funding
SafeSat	1st	2	Support
NewSat	1st	2	Funding
SafeSat Technological 1	2nd	2	Tech
BigSat Funding 1	2nd	2	Funding
BigSat Funding 2	2nd	2	Funding
LittleSat Funding 2	2nd	1.5	Funding
LittleSat Requirements 1	2nd	1.5	Requirements
LittleSat Requirements 2	2nd	1.5	Requirements
SafeSat Funding 1	2nd	1	Funding

The average direct interdependency strength is calculated to be 3, the average second-degree strength is 2.067, and the average of all interdependencies is 2.44. The maximum interdependency is Level 4 – Mandatory, but now there are two instances, one for the BigSat program (Technological) and one for UltraSat (funding). The updated interdependency measurement with changes compared to the first measurement, is summarized as in Table 21. See Appendix B for additional tables and calculations.

Table 21. StarSat program dependency summary two years from program start

Time	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
Program Start	26	2	2.36	0.7	4	1
Two Years Later	25	2	2.44	0.729	4	2
Change	-1	0	0.08	0.029	0	1

The StarSat program interdependency characteristics have changed somewhat in the two years since the beginning of the program. The general level of dependency strength has increased. Table 20 shows that the new UltraSat program has very strong interdependency ties to StarSat based on funding and support. Several other program interdependencies, such as those for EagleSat and NovaSat, have been retired.

Continuing to measure program interdependencies throughout the life of the program will ensure that the program manager has an up-to-date knowledge of how the program may be influenced by other programs. As programs change, program interdependencies change with them. Maintaining current dependency and

interdependency measurements as summarized in Tables 20 and 21 will help ensure that mitigations and protection strategies for adverse interdependence effects are kept up-to-date. Again, these two summary tables are most useful in assessing program dependency and vulnerabilities to interdependency effects

StarSat Criticality

So far we have focused on identifying and measuring program dependencies for the StarSat program. Now that the program dependencies have been measured we will look more closely at the programs which are themselves dependent upon the StarSat program.

There are two possible methods to measure program criticality. The first is to attempt to measure the dependencies of other programs by simply applying the MCIMM model ~~–in~~ reverse," as discussed in Chapter III. We will call this the manual method. The second method is to have all other program managers measure their programs for dependencies using the MCIMM model and see which ones identify StarSat as a depended-upon program.

The first method can be difficult to apply because it is limited by the measurer's knowledge of other programs. The second method is currently difficult to apply because no tool exists at this time to collect, integrate, and analyze those data. However, such a tool could be developed that would automate this process. This method will be referred to as the Automated Method.

For this example, we will use the manual method, which is that of estimating the dependencies of other programs upon StarSat ourselves. We will apply each of the

Interdependency Factors, looking for programs which depend upon StarSat. The Funding and Support factors are fairly straightforward because, as indirect factors, they are two-way. That is, they measure interdependencies rather than dependencies. However, the strength of the interdependency each way may be different. The Technological Factor is more difficult because it is generally one-way. The Systems Interaction Requirements factor may show both dependencies and interdependencies and so may be one- or two-way.

For this example we will limit the scope of the criticality measurement to first-degree dependent programs.

Funding Criticality

The same programs which StarSat depends upon with respect to the funding depend upon StarSat. At program start, these are the EagleSat and NovaSat programs, and two years later they are the UltraSat and NewSat programs. The difference in terms of program criticality vs. program dependency lies in the program priorities. At program start, all three programs are of equal priority. Thus, StarSat has a Level 3 – Dependent criticality to the EagleSat and NovaSat programs, which matches its own Level 3 dependencies upon those programs. StarSat funding criticality is shown in Table 22:

Table 22. StarSat Funding Criticality at program start

Program\Program depended upon	StarSat
EagleSat	3
NovaSat	3

However, two years later, the programs have different priorities and consequently the dependency and criticality levels do not necessarily match. We determined that StarSat has a Level 4 – Mandatory dependency upon UltraSat and a Level 2 – Associated dependency upon NewSat. However, the reverse is not true. UltraSat, being a higher-priority program, is less dependent upon StarSat. It would only have a Level 2 – Associated dependency upon StarSat. If an adverse effect were to occur on the StarSat program, the UltraSat mission would most likely be unaffected. Resources would likely not be pulled from UltraSat to assist StarSat. For NewSat, the opposite is true because it is lower priority. If an adverse effect were to occur to StarSat, it is possible that resources would be moved from the NewSat program, affecting cost, schedule and possibly performance. So while StarSat has a Level 2 – Associated dependency upon NewSat, the NewSat program has a Level 3 – Dependent dependency upon StarSat.

The criticality of the StarSat program with respect to the Funding Interdependency Factor two years from program start is summarized in Table 23:

Table 23. StarSat funding criticality two years from program start

Program\Program depended upon	StarSat
UltraSat	2
NewSat	3

Technological Criticality

To evaluate StarSat criticality from a technological standpoint, we must examine any new technologies, processes, materials, or data being developed internally by StarSat.

These include the SSP payload and the rapid development process. At program start, no other programs were identified which depended upon these technologies. The same is true two years later. StarSat is not critical to any other programs from a technological standpoint.

This is not necessarily a weakness of the StarSat program. In fact, if the StarSat technologies have not been developed to a sufficient Technology Readiness Level (TRL), then it would not be good for other programs to try to incorporate those technologies early on. As the technologies are matured and proven through the StarSat program, they may be more likely to be adopted by other programs.

Support Criticality

Like the Funding Interdependency Factor, the Support factor is fairly simple to evaluate because the interdependencies identified earlier are two-way. All programs upon which StarSat depends with respect to the Support Interdependency Factor depend on StarSat. The degree to which they depend on StarSat may not match the degree to which StarSat depends upon them. For example, BigSat and UltraSat are high-priority programs. Their dependency levels upon StarSat are subsequently lower and would only be Level 2 – Associated rather than the Level 3 dependency of StarSat upon them. Likewise, at two years into the program, SafeSat has a Level 3 dependency upon StarSat, even though at that point StarSat only has a Level 2 dependency upon SafeSat. Program priority should be considered when examining program criticality. Table 24 shows the StarSat program criticality links at program start and at two years into the program:

Table 24. StarSat Support criticality at program start and at two years later

Program\Program depended upon	StarSat (Start)	StarSat (Two Years)
EagleSat	3	-
NovaSat	3	-
BigSat	2	2
LittleSat	3	3
QuickSat	3	3
SV Lab 1	3	3
SV Lab 2	3	3
SafeSat	3	3
RLG-1	2	-
UltraSat	-	2
BrightSat	-	3
FarSat	-	3

Systems Interaction Requirements Criticality

To evaluate program criticality with respect to the Systems Interaction Requirements Interdependency Factor, we look at current acquisition programs for systems which will interact with or depend upon the StarSat system operationally. Note that this does not include fielded systems, only systems still in acquisition. The reason is that we are measuring program criticality rather than systems criticality.

For StarSat, there are no systems under development which will depend upon the system operationally. All systems which will interact with StarSat have already been fielded. (In fact, the StarSat system was specifically designed to integrate with legacy

systems and not to depend on new systems.) At this time, StarSat is not critical to any other acquisitions *programs* because all systems with which StarSat will interact are already in service.

It is critical to understand that this factor does *not* reflect the operational importance of the program being evaluated. The Systems Interaction Requirements Interdependency Factor is only a means to determine if there are other acquisitions *programs* which are dependent upon or interdependent with a certain program. This factor can help find and evaluate these programmatic dependency links. It does not measure the degree to which other *systems* depend upon the program, and it is not a measure of operational criticality or usefulness. The Systems Interaction Requirements Interdependency Factor does *not* measure operational importance. It simply helps to identify dependent or interdependent acquisition programs.

Second-Degree Programmatic Criticality

Once we have determined the first-degree programs which depend upon StarSat, we can evaluate the programs with second-degree or higher dependencies on StarSat. This evaluation is done by applying the criticality measurement method to each of the first-degree dependent programs in order to identify the programs that depend upon them. StarSat is critical to these newly identified programs with a second-degree criticality.

We will not measure second-degree criticality in this example, but it is important to be aware that a given program may still be highly critical to other programs separated by several degrees of dependency. It may be appropriate to account for third, fourth, or

even higher degrees in order to accurately determine how critical a given program is to other acquisition programs within the DoD.

Programmatic Criticality Measurement

We can now integrate the program criticality measurements from the four Interdependency Factors to determine the programmatic criticality of StarSat to other acquisitions programs. As stated previously, this is only a measure of *programmatic* criticality. It does not reflect *systematic* or *operational* criticality. It is a tool to help determine how an acquisition program may impact other acquisition programs.

Tables 25 and 26 summarize the program criticality measurements for StarSat:

Table 25. StarSat first-degree program criticality

Program	Dependency Strength (Program Start)	Dependency Strength (Two Years)	Dependency Factor(s)
EagleSat	3	-	Funding, Support
NovaSat	3	-	Funding, Support
LittleSat	3	3	Support
QuickSat	3	3	Support
SV Lab 1	3	3	Support
SV Lab 2	3	3	Support
SafeSat	3	3	Support
NewSat	-	3	Support
BrightSat	-	3	Support
FarSat	-	3	Support
RLG-1	2	-	Support
BigSat	2	2	Support
UltraSat	-	2	Support

Table 26. StarSat program criticality summary

Time	Number of Dependent Programs	Degrees of Criticality measured	Average Criticality Strength	Std Dev of Criticality Strength	Maximum Criticality Strength	Number of Maximums
Program Start	9	1	2.778	0.416	3	7
Two Years Later	10	1	2.8	0.4	3	8

This example demonstrates program criticality measurement for the StarSat program. These programs would be affected at least to some degree if StarSat were disrupted. With this knowledge, we can see the importance of protecting StarSat against adverse effects, not just for its own sake, but also for the sake of the programs which depend upon it.

Limitations of the Criticality Measurement

The manual method used above maybe subject to bias if program personnel are the ones executing the measurement. Personnel may wish to make the program seem more critical than it actually is. One possible way to avoid this bias is to have a disinterested third party apply the model to take the criticality measurement for the program. However, this may not always be possible.

The automated method is not subject to this bias. This method would collect program dependency data and integrate and analyze it for specific measurements. This method could be integrated into existing DOD program management tools. If this were to be done, it would provide a complete picture of program dependency and interdependency within the DoD, including higher-degree dependency measurements and

program criticality measurement. Program managers would have to evaluate the first-degree dependency and interdependency relationships for their programs. These first-degree evaluations could all be integrated to determine the full scope of dependency or criticality for any desired program.

At this time no specific tool exists for this task. However, a suitable database tool could easily be created which would allow program managers to input program dependency data and would then integrate that data to create a complete and accurate dependency and/or criticality measurement. This idea is discussed as an area for future study in Chapter V.

Future prediction of Interdependency

We can also use the model to predict future interdependency relationships for the program based upon expected progress and changes to the programs. For example, if programs that share funding interdependencies are expected to be complete or fielded within a certain timeframe, then we may be able to predict a reduction in interdependency links for that future timeframe. Likewise, if we expect new programs to enter development which will share funding or support sources, we can make interdependency predictions based on expected program parameters. Also, if we expect new programs to emerge which will depend upon our program, then we can make a prediction of future program criticality.

This is an example of a possible way to use the MCIMM Model. While this particular concept may be useful, validation and verification of this application is beyond the scope of this thesis.

Summary

This chapter presents a simplified yet appropriate application of the interdependency measurement model to an acquisition program. We have been able to demonstrate dependency and criticality measurement to include higher-degree interdependency relationships. We have shown how the model can help identify specific vulnerabilities of a program based on programmatic dependencies and interdependencies. We have also shown how program interdependency can change with time as programs progress and new programs evolve. Finally, we have demonstrated a measure of program criticality by using the model to determine how the program is depended upon by other programs. We have pointed out the potential bias and difficulty of using this method of criticality measurement and have suggested a possible alternate method which may be more accurate.

V. Conclusions and Recommendations

Chapter Overview

This chapter presents the conclusions of the research and its significance and application in DoD program management. It also provides recommendations for directions for future research in the area of program dependency and interdependency measurement.

Conclusions of Research

The MCIMM model can be used to measure program interdependency characteristics for a DoD acquisition program. These measurements can show us the ways in which DoD programs may impact each other. The model can be used to account for higher-degree interdependencies or dependencies with programs which are not directly connected. These higher-degree dependencies can be strongly connected to a program and can cause severe program effects. The MCIMM model provides a way to capture these interdependencies allowing program managers to protect against possible vulnerabilities.

Significance of Research

Until this time there has not been an adequate model for use in measuring program interdependencies. The MCIMM model is the first maturity model to be used to measure program dependency and interdependency. The maturity model concept is well-suited to this application because it can provide an accurate and quantitative measurement

of program interdependency. By using the levels within the MCIMM model, program managers can fully and accurately characterize program interdependency qualities.

Recommendations for Action

The MCIMM should be used to measure program interdependencies within DoD acquisitions. With this model and the accompanying understanding of interdependency impacts, programmatic decisions can be analyzed in the larger context of their effects on other programs. Programs can also be better protected from interdependent effects if program interdependencies are better understood. The MCIMM model provides the method to reach that understanding.

Investigation should also be made into implementation of the automated interdependency and criticality measurement method outlined in the previous chapter. The implementation of this method would allow DoD program managers to make the most effective use of the MCIMM model and could provide acquisition leaders with a full and complete understanding of all program interdependencies within the DoD.

Implementation of this method could require program managers to enter first-degree program dependencies, including dependency strengths, into an online, database application. All program first-degree program dependency data would be stored on a shared database. If the first-degree data for all DoD acquisition programs is correctly entered, then any degree of dependency for any DoD program could be calculated automatically. For example, in order to calculate a second-degree dependency for Program A, we must know the dependencies for a first-degree program, Program B. If the program manager for Program B has entered the Program B first-degree dependency

data into the database, then the associated second-degree dependencies for Program A may be automatically calculated by the computer application. This calculation is possible because the first-degree dependencies for Program B are the second-degree dependencies for Program A. The program manager for Program A does not have to enter, or even know, the Program A second-degree dependencies because they will have already been entered as first-degree dependencies for other programs. A computer program incorporating Equation 2 from this thesis could then determine the strength of each second-degree program dependency to Program A. This capability exists for any degree of program dependency measurement and would allow automatic calculation of second-, third-, fourth-, and higher-degree dependencies automatically. A notional example of a potential output of such a computer application is shown in Table 27:

Table 27. Notional example of automated interdependency measurement

Program A					
Degree	Dependencies	Mean Strength	Std Dev	Max	# of Max
1st	8	2.8	0.63	4	2
2nd	46	2.21	0.75	4	2
3rd	175	1.51	0.45	3.8	1
4th	454	1.02	0.54	3.3	1
∞	672	1.27718	0.45	4	4

This method could be integrated into current web-based, DoD program management tools or statusing programs. The potential benefits of having an integrated database of all program dependencies and interdependencies for all acquisition programs within the DoD are enormous. Program decisions could be evaluated to determine how

they could potentially affect all other DoD programs to any specified degree of distance. An adverse effect to one program could be mitigated by other programs in advance if the full scope of program interdependencies is known. The value of a tool that would allow application of the MCIMM model to all programs within the DoD, and automatic calculation of high-degree program dependencies, cannot be overstated. The MCIMM model provides a method for measuring program interdependency accurately and quantitatively. Use of a computer-based tool for automatic dependency tracking and calculation may be the best way to effectively implement the MCIMM model in the DoD.

Limitations

The scope of this research has been limited to DoD acquisitions systems. While the research and model may be applicable to other areas, such a civil or corporate program management, demonstration or validation of the model in those areas is beyond the scope of this thesis.

The model only provides a snapshot-in-time measurement of program interdependency. For this reason, it is important that the program office apply the model multiple times during the life of a program. The program manager may decide to measure program interdependency on a time-based interval (monthly, quarterly, annually, etc) or on an event-based interval (at program milestones or upon emergence of a new, interdependent program). At the very least, program interdependency should be measured at the start of a program. The frequency of additional measurements is at the discretion of the program manager, with the understanding that interdependencies can change dramatically over the life of a program.

The model does not explicitly include measures of probability of disruption because of adverse effects to an interdependent program. Rather, the model provides guidelines for evaluating interdependency based on the potential severity of a negative impact in any of the four Interdependency Factors. This restriction is not necessarily a weakness, as the model is designed to increase awareness of potential pitfalls caused by interdependencies with other programs. When strong program interdependencies are identified, the program manager may use an estimate of the probability of disruption when determining how to respond to those interdependencies, whether through resource allocation or through development of other mitigation strategies. The Interdependency Level of a connected program should not be adjusted based on the probability of disruption. The Interdependency Level is only used to determine the possible magnitude of a negative impact, not the probability of that impact. The program manager should use probability to determine how to *respond* to interdependency at a given Interdependency Level.

Finally, the model is limited by measurer's knowledge of other potentially interdependent programs. This thesis establishes guidelines for identifying program interdependencies using the four Interdependency Factors. However, the measurer may be unaware of the existence of a potentially interdependent program. If this is the case, then some, possibly significant, interdependencies may be unaccounted for.

Recommendations for Future Research

This thesis has addressed high-degree interdependencies, meaning interdependencies with programs that are connected thorough intermediate programs.

While this thesis has presented algorithms to account for the strength of these higher-degree program dependencies, further study into the effects of distantly-related programs would be useful. A study of the applicability of algorithms used in social network analysis may also provide insight into the nature of high-degree interdependency relationships.

A study into implementation of the automated interdependency and criticality measurement method would also be beneficial. This could be a software program implemented into existing DoD program management software tools or some database program. A study into the potential requirements, implementation methods, and concepts for use of this program would be extremely useful.

Additional studies into the correlation of specific interdependency metrics with program outcomes would serve to further establish the validity of the model. Such studies have already been done with interdependency measurement factors such as N_e and $S(i,j)$ (Flowe, et al., 2010, Mane & DeLaurentis, 2011), and has helped to establish the importance of program interdependency measurement. Specific study into the correlation between average program interdependency levels, maximum interdependency levels, the number of program interdependencies, and program outcomes may help program managers to make the best use of program interdependency metrics derived from the MCIMM model.

Finally, researchers might seek to determine whether any of the Interdependency Factors are particularly likely to become manifest in the current DoD acquisition environment. Such research may enhance understanding of the nature of program

interdependency. In other words, researchers could seek to determine if certain Interdependency Factors, such as Support or Funding, are generally more widespread than other factors. They could also examine whether certain Interdependency Factors are more likely to manifest in particular types of programs. Research could also seek to determine if there is any difference in the way in which adverse effects originating from different Interdependency Factors propagate.

Summary

We have discussed the significance of this research in measuring DoD acquisition program dependencies and interdependencies. The MCIMM model gives the program manager a powerful tool for measuring program interdependency characteristics and determining the ways in which one program may have impacts upon another. As the acquisitions world continues to expand and becomes increasingly complex, program interdependencies will become more and more important. The MCIMM model provides a structure and methodology for fully evaluating these interdependencies. Use of the MCIMM model will lead to greater awareness of program dependency and interdependency relationships, better –understanding of program management decisions, identification of difficult-to-find program vulnerabilities, and a greater probability of program success in the increasingly interdependent acquisition environment.

Appendix A – Interdependency Metrics Equations

The following is a summary of the measurement equations used with the MCIMM model:

Number of Dependencies

The number of dependencies is determined by counting the number of programs having dependency connections to the program being measured. It is an integer value. In this thesis we have counted the number of *programs* which are dependent or depended-upon, rather than the number of dependency *links* or *connections*. This is important because a single program may have multiple dependency links. For example, a single program may be connected through both Funding and Technological dependencies. Therefore, the number of connected programs and the number of connections, or links, could be very different.

The number of dependencies is calculated by using the Interdependency Factors presented in the MCIMM to determine the number of programs with dependency connections to the program being measured.

Strength of Dependency

The strength of a first-degree dependency is found by applying the Interdependency Levels in the MCIMM model. The dependency is evaluated against the criteria established for each level of the model. When the appropriate level is

determined, the dependency is assigned a numerical value commensurate with that level. The value will be between 0 and 4.

Degree of Interdependency

The degree of interdependency is as the number of steps between a program and the program being measured. If a program can directly influence another program then it has a first-degree dependency connection. A program which can influence another program by affecting an intermediate program has a higher-degree dependency.

The maximum degrees of dependency measured are determined when setting the scope of the measurement. It may be appropriate to measure only first-degree connections or it may be appropriate to measure much higher degrees. As the degree of dependency increases, the strength of the dependency connection to the original program generally becomes weaker. However, higher-degree dependencies can often have significant effects to a program.

Higher-Degree Dependency Strength

Higher-degree dependencies are measured in a two-step process. First, the strength of their dependency with an immediately connected program is measured. For example, for a second-degree program, the dependency connection with a first-degree program would be measured. The dependency with the original program is then calculated using Equation 2, repeated here:

$$S_{0,x} = 0.25^{x-1}\left(S_{0,1} * \cdots * S_{x-1,x}\right)$$

Average Dependency Strength

When all program dependencies have been measured, the average dependency strength $\bar{S}$, can be calculated. This value is the mean of all measured dependency values. The formula for this calculation is given by Equation 3 below:

$$\bar{S} = \frac{1}{N}(S_1 + \cdots + S_N)$$

Equation 3. Average interdependency strength $\bar{S}$

In this equation, *N* represents the number of dependencies measured, and $S_1 + \cdots + S_N$ are the individual strengths of each dependency

Standard Deviation of Dependency Strength

The standard deviation of dependency strength can tell us how closely the different dependency strengths are gathered around the mean value. Large standard deviations mean a greater variety of dependency strengths. Smaller values mean that most dependency strengths are close to the mean value.

This standard deviation of dependency strength is calculated with Equation 4.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i - \bar{S})^2}$$

Equation 4. Standard deviation of dependency strength

For this equation, N again represents the number of dependencies measured, S_i is the individual strength of each dependency, and $\bar{S}$ is the average dependency strength.

Maximum Dependency Strength

The maximum dependency strength is determined by examining the dependencies measured. The maximum strength indicates the most significant potential effects to the program based on interdependencies.

Number of Occurrences of the Maximum Strength

The number of occurrences of the maximum strength is also determined by inspection. This is the number of programs that manifest the highest level of interdependency observed, and therefore have the potential for the most significant, or most severe, interdependent effects.

Summary

These are a few of the metrics that can be obtained using the MCIMM model and are the metrics used in the examples in this thesis. Table 28 provides a summary of these metrics with their definitions, uses, and methods of calculation.

Table 28. MCIMM metric summary table

Metric	What it is	How to Calculate it	What it Tells Us
Number of Dependencies	The number of dependencies up to or at a certain degree	Sum the number of programs with dependency links (NOT the number of links)	How many programs have a dependency relationship with the program being measured
Degrees Measured	The magnitude of separation of the dependencies measured	Predetermined by the scope of the measurement	The extent to which distant dependencies have been measured
Average Dependency Strength	The average level of dependency connection for the program.	$\bar{S} = \frac{1}{N}(S_1 + \cdots + S_N)$	The general level of dependency strength for the program
Std Dev of Dependency Strength	The closeness of all dependencies in general to the average value	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i - \bar{S})^2}$	How close all dependencies are to the average value
Maximum Dependency Strength	The level of dependency of the strongest dependency connection for the program	Largest value of S_i in the set S	The strength of the strongest dependency links for the program being measured
Number of Maximums	The number of programs exhibiting the maximum level of dependency	Number of occurrences of the largest value of S_i	The number of programs having the strongest dependency links

Appendix B – StarSat Interdependency Calculations

The StarSat program was measured for first- and second-degree program dependencies at program start and at a point in time two years later. The program was also measured to determine first-degree program criticality at the same times. These measurements were presented in Chapter IV. This appendix shows the calculations used to determine those measurements.

For the example case we presented six metrics to characterize the dependencies of the StarSat program. These were: the number of dependencies, the degree of dependency measured, the average strength of the dependencies, the standard deviation of the dependencies, the maximum strength of the dependencies, and the number of occurrences of the maximum strength. The calculations of these metrics are presented here.

First-Degree Dependencies at Program Start

The MCIMM measurement process identified ten first-degree dependencies for StarSat using the Interdependency Factors. The strengths of those dependencies were then measured using the Interdependency Levels. The results are summarized in Table 29.

Table 29. StarSat first-degree dependencies summary

Program	Dependency Strength	Dependency Factor(s)
BigSat	4	Tech, Support
NovaSat	3	Funding, Support
EagleSat	3	Funding, Support
SV Lab A	3	Support
SV Lab B	3	Support
SafeSat	3	Support
QuickSat	3	Support
LittleSat	3	Support
RLG 1	2	Support
Solar Panel Initiative	2	Tech

Number of Dependences

For the StarSat first-degree measurement at program start, ten programs were identified upon which StarSat is dependent. The value of this metric then, is 10.

Degrees of Dependency Measured

This value is set when determining the scope of the measurement. At this point the value is 1 because we have only measured first-degree dependencies.

Average Strength of the Dependencies

Table 25 presents the strength of each dependency. To determine the average value of the dependency strengths, $\bar{S}$, we apply Equation 3, where $N = 10$ and S_i is the strength of each dependency.

We than have the following:

$$\bar{S} = \frac{1}{10}(4 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 2 + 2) = 2.9$$

Standard Deviation of Dependencies

With the average dependency strength calculated, we can find the standard deviation of the dependency strengths. This is calculated with Equation 4.

$$\sigma = \sqrt{\frac{1}{10}\{(4 - 2.9)^2 + [7(3 - 2.9)^2] + [2(2 - 2.9)^2]\}} = 0.539$$

Maximum Strength of Dependencies

Examination of the first-degree dependencies as shown in Tables 8 and 25, shows that the maximum dependency strength is Level 4 – Mandatory. The value of this metric is 4.

Number of Occurrences of the Maximum Strength

Again, by examining Tables 8 and 25, we can see that there is one instance of the maximum dependency strength. The value for the metric is then 1. This dependency manifests with the BigSat program.

First-Degree Dependency Summary at Program Start

With the metrics calculated as above we can summarize the first-degree program dependencies. This summary was shown in Table 12 and is repeated here.

Table 12. StarSat First-Degree dependency summary

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	10	1	2.9	0.539	4	1

These metrics show the general characteristics of the StarSat first-degree dependencies. We can see that there is one highly critical program with most other programs being at Level 3. This effectively shows us that most of the first-degree, depended-upon programs at program start could have significant effects to StarSat.

Second-Degree Dependencies at Program Start

The second-degree programs were measured in much the same way as the first degree programs, but with an additional step. The MCIMM model was applied to each of the first-degree programs to identify additional, second-degree program dependencies (See Appendix C), and to determine the strengths of these second-degree programs upon the first-degree programs. The unique second-degree programs, and the dependency strengths of the first-degree programs, are summarized in Table 9, repeated here. The additional step is to determine the strength of each second-degree dependency with StarSat using Equation 2.

Table 9. List of StarSat second-degree dependencies

First-Degree Program	Second-Degree Program	Dependency strength (for First-Degree Program)
BigSat	BigSat Funding 1	2
	BigSat Funding 2	2
LittleSat	LittleSat Funding 1	3
	LittleSat Funding 2	2
	LittleSat Requirements 1	2
	LittleSat Requirements 2	2
	LittleSat Requirements 3	3
QuickSat	QuickSat Funding 1	3
	QuickSat Requirements 1	4
	QuickSat Requirements 2	4
SafeSat	SafeSat Funding 1	2
	SafeSat Technological 1	4
RLG-1	RLG Funding 1	3
	RLG Funding 2	3
	RLG Requirements 1	4
	RLG Requirements 2	3

Second-Degree Dependency Strengths

Once the second-degree programs have been identified, we can calculate the strength of each second-degree dependency with Equation 2. The calculations for each second-degree program are shown below. For second-degree calculations we set $x = 2$.

BigSat Funding 1: $S_{1,2} = 0.25^{1}(4 * 2) = 2$

BigSat Funding 2: $S_{1,2} = 0.25^{1}(4 * 2) = 2$

LittleSat Funding 1: $S_{1,2} = 0.25^{1}(3 * 3) = 2.25$

LittleSat Funding 2: $S_{1,2} = 0.25^1(3 * 2) = 1.5$

LittleSat Requirements 1: $S_{1,2} = 0.25^1(3 * 2) = 1.5$

LittleSat Requirements 2: $S_{1,2} = 0.25^1(3 * 2) = 1.5$

LittleSat Requirements 3: $S_{1,2} = 0.25^1(3 * 3) = 2.25$

QuickSat Funding 1: $S_{1,2} = 0.25^1(3 * 3) = 2.25$

QuickSat Requirements 1: $S_{1,2} = 0.25^1(3 * 4) = 3$

QuickSat Requirements 2: $S_{1,2} = 0.25^1(3 * 4) = 3$

SafeSat Funding 1: $S_{1,2} = 0.25^1(3 * 2) = 1.5$

SafeSat Technological 1: $S_{1,2} = 0.25^1(3 * 4) = 3$

RLG Funding 1: $S_{1,2} = 0.25^1(2 * 3) = 1.5$

RLG Funding 2: $S_{1,2} = 0.25^1(2 * 3) = 1.5$

RLG Requirements 1: $S_{1,2} = 0.25^1(2 * 4) = 2$

RLG Requirements 2: $S_{1,2} = 0.25^1(2 * 3) = 1.5$

Now that these dependency strengths have been determined we can measure the other metrics.

Number of Dependences

For the StarSat second-degree measurement at program start, 16 unique programs were identified upon which first-degree programs are dependent. The value for this metric is 16.

Note that in Appendix C, many more second-degree programs are identified for the firs-degree programs, especially for the Support Factor. However, these programs are

already identified as first-degree programs for StarSat. So while QuickSat depends upon BigSat, BigSat is already identified as a strong first-degree dependency for StarSat, and so it is not counted as an additional second-degree dependency.

It is possible though, that a program may have a stronger second-degree effect than its first-degree effect. For example, if BigSat had a Level 4 – Mandatory dependency upon QuickSat, the second-degree dependency value for StarSat upon QuickSat would be Level 4 – Mandatory, as well. This is higher than the first-degree dependency value which is only Level 3 – Dependent. While this is not the actual case for the StarSat example, the program office should be aware of the potential that second-degree dependencies may be just as important as, or even more important than, first-degree dependencies.

Degrees of Dependency Measured

We ware only measuring a single degree of dependency: the second-degree dependencies. Because only one degree is being measured we note this as ~~–2~~nd only."

Average Strength of the Dependencies

We have calculated the strength of each second-degree dependency already. We again use Equation 3 to determine the average value of the second-degree dependency strengths. For this case, $N = 16$ because we have 16 second-degree dependencies.

$$\bar{S} = \frac{1}{16}(2 + 2 + 2.25 + 1.5 + 1.5 + 1.5 + 2.25 + 2.25 + 3 + 3 + 1.5 + 3 + 1.5 + 1.5 + 2 + 1.5) = 2.016$$

Standard Deviation of Dependencies

With the average dependency strength calculated, we can find the standard deviation of the dependency strengths. This is calculated with Equation 4.

$$\sqrt{\frac{1}{10}\{[3(3-2.016)^2]+[3(2.25-2.016)^2]+[3(2-2.016)^2]+[7(1.5-2.016)^2]\}}$$

$$= 0.56$$

Maximum Strength of Dependencies

The maximum strength of the second-degree dependencies for StarSat is 3.

Number of Occurrences of the Maximum Strength

There are three instances of the maximum dependency strength. These are for systems interaction requirements dependencies for LittleSat and a Technological dependency for SafeSat.

Second-Degree Dependency Summary at Program Start

With the metrics calculated as above we can summarize the second-degree program dependencies as shown in Table 13 (repeated here).

Table 13. StarSat second-degree dependency summary

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	16	2nd only	2.016	0.56	3	3

These metrics show the general characteristics of the StarSat second-degree dependencies at program start. We can see that there are several other significant programs which may have been missed had we confined measurement to first-degree dependencies only.

Total Dependency Calculations for StarSat at Program Start

With both the first and second degree dependencies measured we can summarize the total StarSat program dependency measurements at program start.

Number of Dependences

The total number of program dependencies is simply the sum of the number of dependencies for each degree measured. In this case we have 10 first-degree dependencies and 16 second-degree dependencies. The total number of dependencies at the start of the program is 26.

Degrees of Dependency Measured

We have measured first and second degree dependencies. The value for this metric is then 2.

Average Strength of the Dependencies

We can combine the average strength of the first- and second-degree dependencies to find the total average strength. There are 10 first-degree dependencies with an average strength of 2.9, and 16 second-degree dependencies with an average

strength of 2.016. The total average dependency strength is can then be calculated with Equation 3.

$$\overline{S_{tot}} = \frac{1}{26}[10(2.9) + 16(2.016)] = 2.36$$

Standard Deviation of Dependencies

With the total average dependency strength calculated, we can find the standard deviation of the dependency strengths.

$$\sigma = \sqrt{\frac{1}{10}\{(4-2.36)^2 + [10(3-2.36)^2] + [3(2.25-2.36)^2] + [5(2-2.36)^2] + [7(1.5-2.36)^2]\}}$$

$$= 0.7$$

Maximum Strength of Dependencies

The maximum dependency strength for all program dependencies at program start is Level 4 – Mandatory.

Number of Occurrences of the Maximum Strength

There is one instance of the maximum dependency strength. This dependency is on the BigSat program.

Total Program Dependency Summary at Program Start

We can summarize the total program dependencies to include first- and second-degree dependencies. This summary was shown in Table 14 and is repeated here.

Table 14. StarSat total program dependency summary at program start

Program	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
StarSat	26	2	2.36	0.7	4	1

Calculations for Dependencies Two Years after Program Start

The same steps detailed above are used to calculate the program dependency characteristics two years into the program. We will focus on the total dependency measurement rather than the overall averages and metrics for first- and second-degree program dependencies separately.

We must still calculate the individual strengths of each dependency starting with the first-degree dependencies. Using the MCIMM model ten first-degree interdependencies were measured. These are shown in Table 30 below.

Table 30. StarSat first-degree dependences two years after program start

Program	Dependency Strength	Dependency Factor(s)
BigSat	4	Tech, Support
UltraSat	4	Funding, Support
SV Lab A	3	Support
SV Lab B	3	Support
QuickSat	3	Support
LittleSat	3	Support
FarSat	3	Support
BrightSat	3	Support
SafeSat	2	Support
NewSat	2	Funding

Second-Degree Dependency identification

The second-degree dependencies were identified and measured as before. Appendix C for shows the dependency strengths between the first- and second degree programs.

Number of Dependences Two Years Later

For the StarSat dependency measurement two years after program start we identified ten first-degree programs as shown in Table 29, and 15 second-degree programs as shown in Appendix C. The value of this metric is 25.

Degrees of Dependency Measured

This value is set when determining the scope of the measurement. For this measurement the value is 2 because we are measuring first-and second-degree dependences.

Second-Degree Dependency Strengths

With the updated dependency measurements the new second-degree program dependencies can be calculated. The equations are given below. We will only show those second-degree dependencies which are new are different from the previous measurement. Note that the second-degree dependencies associated with SafeSat are recalculated. This is because of the change in the SafeSat first-degree dependency level from Level 3 to Level 2.

SafeSat Funding 1: $S_{1,2} = 0.25^1(2 * 2) = 1$

SafeSat Technological 1: $S_{1,2} = 0.25^1(2 * 4) = 2$

FarSat Funding 1: $S_{1,2} = 0.25^1(3 * 3) = 2.25$

BrightSat Funding 1: $S_{1,2} = 0.25^1(3 * 3) = 2.25$

BrightSat Funding 2: $S_{1,2} = 0.25^1(3 * 3) = 2.25$

Now that these dependency strengths have been determined we can measure the other metrics.

Average Strength of the Dependencies

Table 20 presents the strength of each dependency. It is repeated here. To determine the average value of the dependency strengths, $\bar{S}$, we apply Equation 3, where $N = 25$ and S_i is the strength of each dependency. Note that rather than sum each individual dependency, we multiply by each level of dependency strength by the number of times it occurs. For example, there are two Level 4 dependences. We represent this with 2(4) rather than 4+4.

We than have the following equation for the average dependency strength two years after program start:

$$\bar{S} = \frac{1}{25}[2(4) + 8(3) + 6(2.25) + 5(2) + 3(1.5) + 1] = 2.44$$

Table 20. StarSat program dependencies two years after program start

Program	Degree	Dependency Strength	Dependency Factor(s)
BigSat	1st	4	Tech, Support
UltraSat	1st	4	Funding, Support
SV Lab A	1st	3	Support
SV Lab B	1st	3	Support
QuickSat	1st	3	Support
LittleSat	1st	3	Support
FarSat	1st	3	Support
BrightSat	1st	3	Support
QuickSat Requirements 1	2nd	3	Requirements
QuickSat Requirements 2	2nd	3	Requirements
LittleSat Funding 1	2nd	2.25	Funding
LittleSat Requirements 3	2nd	2.25	Requirements
QuickSat Funding 1	2nd	2.25	Funding
BrightSat Funding 1	2nd	2.25	Funding
BrightSat Funding 2	2nd	2.25	Funding
FarSat Funding 1	2nd	2.25	Funding
SafeSat	1st	2	Support
NewSat	1st	2	Funding
SafeSat Technological 1	2nd	2	Tech
BigSat Funding 1	2nd	2	Funding
BigSat Funding 2	2nd	2	Funding
LittleSat Funding 2	2nd	1.5	Funding
LittleSat Requirements 1	2nd	1.5	Requirements
LittleSat Requirements 2	2nd	1.5	Requirements
SafeSat Funding 1	2nd	1	Funding

Standard Deviation of Dependencies

With the average dependency strength calculated, we can find the standard deviation of the dependency strengths. This is calculated with Equation 4.

$$\sigma = \sqrt{\frac{1}{25}\left\{\begin{matrix}[2(4-2.48)^2]+[8(3-2.48)^2]+[6(2.25-2.48)^2]+\\ [5(2-2.48)^2]+[3(1.5-2.48)^2]+(2.25-2.48)^2\end{matrix}\right\}} = 0.729$$

Maximum Strength of Dependencies

Examination of the first- and second-degree dependencies as shown in Table 20 shows that the maximum dependency strength is Level 4 – Mandatory. The value of this metric is 4.

Number of Occurrences of the Maximum Strength

We also note that there are two occurrences of Level 4 dependency. There are with the BigSat and UltraSat programs. The value for the metric then is 2.

Dependency Summary Two Years after Program Start

With the metrics calculated as above we can summarize the program dependencies. This summary is shown as part of Table 21 and is repeated here:

Table 21. StarSat program dependency summary two years from program start

Time	Number of Dependencies	Degrees of Dependency measured	Average Dependency Strength	Std Dev of Dependency Strength	Maximum Dependency Strength	Number of Maximums
Program Start	26	2	2.36	0.7	4	1
Two Years Later	25	2	2.44	0.729	4	2
Change	-1	0	0.08	0.029	0	1

Criticality Measurement

The StarSat criticality measurement in our example is limited to first-degree criticality. In order to accomplish this measurement, all programs which depended upon StarSat are identified and the strengths of their dependencies measured. This was done in Chapter IV and summarized in Table 25 which is repeated here:

Table 25. StarSat first-degree dependency summary

Program	Dependency Strength (Program Start)	Dependency Strength (Two Years)	Dependency Factor(s)
EagleSat	3	-	Funding, Support
NovaSat	3	-	Funding, Support
LittleSat	3	3	Support
QuickSat	3	3	Support
SV Lab 1	3	3	Support
SV Lab 2	3	3	Support
SafeSat	3	3	Support
NewSat	-	3	Support
BrightSat	-	3	Support
FarSat	-	3	Support
RLG-1	2	-	Support
BigSat	2	2	Support
UltraSat	-	2	Support

Number of Criticalities

At program start, nine other programs depend upon StarSat to some extent. Two years later, ten programs depend upon StarSat. The values for these metrics are 9 and 10.

Degrees of Criticality Measured

For this example we determined to measure only first-degree criticality. The value of the metric is 1.

Average Strength of the Criticalities

Base on Table 25, we can calculate the average criticality strength at program start and two years after program start. We use Equation 3 for this calculation. At program star, $N = 9$. At the two-year timeframe, $N = 10$.

We than have the following equations for the average StarSat criticality strength:

At program start:

$$\bar{S} = \frac{1}{9}(3 + 3 + 3 + 3 + 3 + 3 + 3 + 2 + 2) = 2.778$$

Two years after program start:

$$\bar{S} = \frac{1}{10}(3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 2 + 2) = 2.8$$

Standard Deviation of Criticalities

Now that we have calculated the average criticality strength, we can determine the standard deviation of the criticality strength. This is calculated with Equation 4.

At program start:

$$\sigma = \sqrt{\frac{1}{9}\{[7(3 - 2.75)^2] + [2(2 - 2.75)^2]\}} = 0.416$$

Two years after program start:

$$\sigma = \sqrt{\frac{1}{10}\{[8(3-2.75)^2] + [2(2-2.75)^2]\}} = 0.4$$

Maximum Strength of Criticality

The maximum strength of criticality for StarSat both at program start and two years later is Level 3 – Dependent.

Number of Occurrences of the Maximum Strength

There are six instances of the maximum criticality value at program start and seven instances two years after program start. The values for the metric are then 7 and 8.

Dependency Summary Two Years after Program Start

With the criticality metrics calculated, we can summarize the program the program criticality as below. This summary is shown in Table 26 and is repeated here:

Table 26. StarSat program criticality summary

Time	Number of Dependent Programs	Degrees of Criticality measured	Average Criticality Strength	Std Dev of Criticality Strength	Maximum Criticality Strength	Number of Maximums
Program Start	9	1	2.778	0.416	3	7
Two Years Later	10	1	2.8	0.4	3	8

Appendix C – Second-Degree Dependency Analysis for StarSat

In order to calculate the StarSat second-degree dependencies, the MCIMM model was applied to each first-degree program. The tables following show the analysis used to determine the second-degree interdependencies of the StarSat program. Each first-degree program is analyzed in turn. Table 31 identifies second-degree dependencies at the start of the program, while Table 32 identifies second-degree dependencies two years after the start of the program. Note that the accuracy of this analysis is limited by the analyst's knowledge regarding the first-degree programs and their dependencies. .

The strength of each second degree program dependency for the associated first-degree program is determined using the Interdependency Levels in the MCIMM model.

Once all of the second-degree dependencies have been identified and measured, the unique second-degree dependencies need to be isolated. Notice that for these programs, all of the second-degree Support dependencies have already been accounted for as first-degree dependencies for the StarSat program. They are stronger as first-degree dependencies than they are as second-degree dependencies, and so we do not count them again here. The unique second-degree dependencies are highlighted in Tables 31 and 32.

Once the unique second degree dependencies have been indentified and their dependency strengths to the first-degree programs measured, we can proceed with measuring their second-degree strengths with the StarSat program. This is done using Equation 2. The calculations are shown in Appendix B.

Table 31. Second-degree dependencies at program start

At Program Start	BigSat		LittleSat		QuickSat		SafeSat		EagleSat	
	Program	Dependency Strength	Program	Strength	Program	Strength	Program	Strength	Program	Strength
Funding	BigSat Funding 1	2	LittleSat Funding 1	3	QuickSat Funding 1	3	SafeSat Funding 1	2	NovaSat	3
	BigSat Funding 2	2	LittleSat Funding 2	2	QuickSat Requirements 1	2	SafeSatTechnological 1	3	StarSat	3
					(Same program as below)		(Same program as below)			
Support	LittleSat	2	BigSat	4	BigSat	4	BigSat	4	BigSat	2
	QuickSat	2	QuickSat	3	LittleSat	2	LittleSat	3	QuickSat	1
	SafeSat	2	SafeSat	3	SafeSat	2	QuickSat	4	SatSat	1
	EagleSat	2	NovaSat	3	EagleSat	2	EagleSat	1	NovaSat	3
	NovaSat	2	StarSat	3	NovaSat	3	NovaSat	3	StarSat	3
	StarSat	2			StarSat	3	StarSat	3		
Requirements			LittleSat Requirements 1	2	QuickSat Requirements 1	4	SafeSatTechnological 1	4		
			LittleSat Requirements 2	2	QuickSat Requirements 2	4	(Same program as below)			
			LittleSat Requirements 3	3						
Technological			BigSat	4	BigSat	4	SafeSatTechnological 1	4		
							BigSat	4		

At Program Start	NovaSat		RLG 1		SV Lab 1		SV Lab 2		Solar Panel Initiative	
	Program	Strength	Program	Strength	Program	Strength	Program	Strength	Program	Strength
Funding	EagleSat	3	RLG Funding 1	3						
	StarSat	3	RLG Funding 2	3						
			RLG Requirements 1	3						
			RLG Requirements 2	3						
Support	BigSat	3	StarSat	2			None Identified			
	LittleSat	3								
	QuickSat	3								
	SafeSat	3								
	EagleSat	3								
	StarSat	3								
Requirements			RLG Requirements 1	4						
			RLG Requirements 2	3						
			(Same programs as above)							
Technological										

Table 32. Second-degree dependencies two years after the start of the program

Program Start + Two Years	BigSat		LittleSat		QuickSat		SafeSat		UltraSat	
	Program	Strength	Program	Strength	Program	Strength	Program	Strength	Program	Strength
Funding	BigSat Funding 1	2	LittleSat Funding 1	3	QuickSat Funding 1	3	SafeSat Funding 1	2	StarSat	2
	BigSat Funding 2	2	LittleSat Funding 2	2	QuickSat Requirements 1	2	SafeSatTechnological 1	4	NewSat	1
					(Same program as below)		(Same program as below)			
							FarSat	2		
Support	LittleSat	2	BigSat	4	BigSat	4	BigSat	4	BigSat	3
	QuickSat	2	QuickSat	3	LittleSat	2	LittleSat	3	LittleSat	2
	SafeSat	2	SafeSat	3	SafeSat	1	QuickSat	4	QuickSat	3
	UltraSat	2	UltraSat	3	UltraSat	3	UltraSat	4	SafeSat	1
	BrightSat	2	BrightSat	3	BrightSat	3	BrightSat	3	BrightSat	3
	FarSat	1	FarSat	2	FarSat	2	FarSat	2	FarSat	2
	StarSat	2	StarSat	3	StarSat	2	StarSat	3	StarSat	2
Requirements			LittleSat Requirements 1	2	QuickSat Requirements 1	4	SafeSatTechnological 1	4		
			LittleSat Requirements 2	2	QuickSat Requirements 2	4	(Same program as below)			
			LittleSat Requirements 3	3						
Technological			BigSat	4	BigSat	3	SafeSatTechnological 1	4		
							BigSat	4		

Program Start + Two Years	BrightSat		NewSat		FarSat		SV Lab 1		SV Lab 2	
	Program	Strength	Program	Strength	Program	Strength	Program	Strength	Program	Strength
Funding	BrightSat Funding 1	3	UltraSat	4	FarSat Funding 1	3				
	BrightSat Funding 2	3	StarSat	3						
Support	BigSat	3			BigSat	4	None Identified			
	LittleSat	3			LittleSat	3				
	QuickSat	3			QuickSat	3				
	SafeSat	2			SafeSat	3				
	UltraSat	3			UltraSat	4				
	FarSat	2			BrightSat	2				
	StarSat	2			StarSat	3				
Requirements										
Technological					BigSat	4				

Appendix D – Checklist for Applying the MCIMM Model

Step 1: Determine the Scope of the Measurement

To what degree of dependency will we measure?

Will the measurement be confined to a certain office or agency or will it encompass all DoD programs?

Step 2: Use the MCIMM Interdependency Factors to Identify Program Dependencies

Funding: Where does the money for the program come from and who else gets money from that source?

Technological: What programs are developing something new that is needed for our program?

Support: What other programs are supported by the agencies, organizations, contractors, or other entities responsible for supporting our program?

Systems Interaction Requirements: What in-development systems will our system depend upon operationally? What in-development systems will depend upon us?

Step 3: Use the Interdependency Levels to Measure the Interdependency Strengths

Step 4: Evaluate Higher-Degree Dependencies as Required

Apply MCIMM model to each lower-degree program

Use Equation 2 to determine dependency strength for higher-degree dependencies.

$$S_{0,x} = 0.25^{x-1}\left(S_{0,1} * \cdots * S_{x-1,x}\right)$$

Step 5: Collect Measurements

Total programs depended upon

Average dependency strength

Standard deviation

Maximum strengths, number of maxima

Other factors as necessary or helpful

Step 6: Take Appropriate Action

Plan mitigations to protect program against interdependent effects

Monitor critical programs for disruptions

Step 7: Reassess Program as Needed

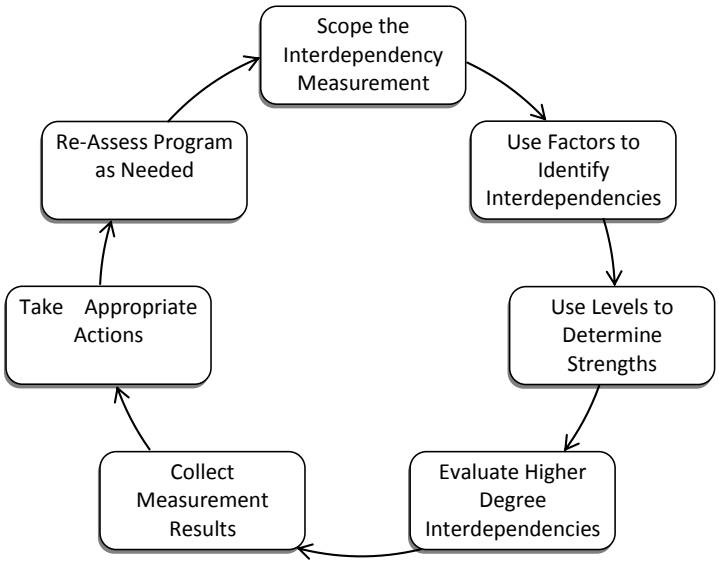
Determine frequency of measurements, when to take measurements

Quarterly

Annually

Monthly

Based on program events



Bibliography

Asikoglu, O., & Simpson, T. W. (September 2010). A New Approach for Evaluating Design Dependencies in Product Architectures. *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference.* Fort Worth, TX.

Brown, M. M., Flowe, R. M., & Hamel, S. P. (2007). The Acquisition of Joint Programs: The Implications of Interdependencies. *CrossTalk The Journal of Defense Software Engineering*, 5.

Brown, M. M., Kravchuk, R., & Owen, G. (2011). Acquisition Risks in a World of Joint Capabilities. *Eighth Annual Acquisition Research Symposium.* Seaside, CA.

Clark, T., & Jones, R. (June 29 - July 1,1999). Organisational Interoperability Maturity Model for C2. *9th International Command and Control Research and Technology Symposium.* Newport, RI.

DeLaurentis, D., & Mane, M. (10 NOV 2010). Assessing the Impact of Development Disruptions in Interdependent Systems. *2010 Annual SERC Research Review.* University of Maryland.

DeLaurentis, D., & Sauser, B. (April 13, 2010). Dynamic Modeling of Programmatic and Systematic Interdependence for System of Systems. *System of Systems Engineering Collaborators Information Exchange (SoSECIE).*

Department of Defense. (1998). *C4ISR Architecture Working Group Final Report - Levels of Information System Interoperability (LISI).* Washington DC: OSD(ASD(C3I)) C4ISR AWG.

dependence. (2011). *Merriam-Webster.com*. Retrieved November 23, 2011, from http://www.merriam-webster.com/dictionary/dependence

Flowe, R. (2007). Capabilities Based Acquisition & Systems of Systems Adapting to a New Paradigm.

Flowe, R. M., Kasunic, M., Brown, M. M., Hardin, P. L., McCurley, J., Zubrow, D., & Anderson, W. B. (2010). *Programmatic and Constructive Interdependence: Emerging Insights and Predictive Indicators of Development Resource Demand.* Software Engineering Institute.

Ford, T. C., Colombi, J. M., & Jacques, D. R. (2009). A General Method of Measuring Interoperability and Describing its Impact on Operational Effectiveness. *Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 16.

http://www.defenseindustrydaily.com. (2005, October 24). Retrieved November 2011, 30, from http://media.defenseindustrydaily.com/images/AIR_E-8_JSTARS_Connectivity_lg.jpg

interdependence, n. (2011, September). *OED Online*. Oxford University Press. Retrieved November 23, 2011, from OED Online: http://www.oed.com.libproxy.wustl.edu/view/Entry/97699

Mane, M., & DeLaurentis, D. (30 April 2011). Capability and Development Time Trade-off Analysis in Systems-of-Systems. *Eighth Annual Acquisition Research Symposium.* SeaSide, CA.

Mane, M., & DeLaurentis, D. D. (2009). Impact of Programmatic System Interdependencies on System-of-Systems Development. *System of Systems Engineering, 2009. SoSE 2009. IEEE International Conference on*, (pp. 1-6). Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5282326&isnumber=5282304

Shibata, J. T. (2010). *A Space Acquisition Leading Indicator Based on System Interoperation Maturity.* Wright-Patterson Air Force Base, OH: Air Force Institute of Technology.

Vita

Captain Matthew B Christensen joined the USAF in 2007, commissioning through Brigham Young University ROTC. Capt Christensen's first assignment on active duty was to a space launch unit where he worked as the Launch Vehicle Mission Manager for the program referred to in this thesis as StarSat. During that assignment, Capt Christensen was able to observe first-hand, the effects of program interdependency.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
December 2011	Master's Thesis	Dec 2010 – Dec 2011

4. TITLE AND SUBTITLE

A METHOD FOR MEASURING PROGRAMMATIC DEPENDENCY AND INTERDEPENDENCY BETWEEN DOD ACQUISITION PROGRAMS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Christensen, Matthew B., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GSE/ENV/11- D02DL

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

OSD/ATL, Attn: Mr. Robert M. Flowe, 4800 Mark Center Dr 15F09-02, Alexandria, VA 22350
Robert.Flowe@osd.mil
571-372-6231

10. SPONSOR/MONITOR'S ACRONYM(S)
OUSD(AT&L)/ARA/EIOS

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

There is currently no accepted method for quantitative measurement of acquisition program interdependency. While some methods exist for measuring dependency or interdependency at the component or system level, these methods do not translate well to program interdependency measurement. The objective of this thesis is to provide a model for measuring acquisition program interdependency accurately and quantitatively.

The model presented in this thesis uses four Interdependency Factors to identify dependency relationships between programs. Specific Interdependency Levels are then used to measure the strengths of those dependencies. The model also accounts for measurement of dependencies upon programs that are not directly connected, i.e., programs that have a degree of separation from another program, and measurement of program criticality, or the extent to which a program is depended-upon.

In this thesis, the measurement model is applied to an example program to measure program dependency characteristics. The results demonstrate that the model can be effectively used to identify and measure program dependencies. The model gives the program manager a quantitative tool to determine how much a program depends upon other programs and the potential impacts of those dependencies. With this information, program managers can better protect against vulnerabilities associated with interdependent effects from other programs.

15. SUBJECT TERMS

ACQUISITION PROGRAM, DEPENDENCY, INTERDEPENDENCY, MEASUREMENT

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Joseph R Wirthlin, Lt Col, USAF
U	U	U	UU	139	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-6565, ext 4650 (Joseph.Wirthlin@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18